НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Обчислювальної техніки

До захисту допущено:

Завідувач кафедри

Сергій СТІРЕНКО

«___»_____20__ р.

Дипломний проєкт

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Комп'ютерні системи та мережі»

спеціальності 123 «Комп'ютерна інженерія»

на тему: «Веб-сервіс для інтелектуального опрацювання документів на основі NER»

Виконав (-ла):
студент (-ка) IV курсу, групи ІО-61
Левківський Вадим Валерійович _____

Керівник:
Викладач, к. т. н., доц. кафедри ОТ,
Болдак Андрій Олександрович _____

Консультант з нормоконтролю:
Професор кафедри ОТ, д.т.н.,
Сімоненко Валерій Павлович _____

Рецензент:
Викладач, к. т. н., доц. кафедри АСОІУ,
Ліщук Катерина Ігорівна _____

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2020 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Обчислювальної техніки

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 123 «Комп'ютерна інженерія»

Освітньо-професійна програма «6.050102 - Комп'ютерні системи та мережі»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Сергій СТІРЕНКО
«___»_____20__ р.

ЗАВДАННЯ
на дипломний проєкт студенту
Левківського Вадима Валерійовича

1. Тема проєкту «Веб-сервіс для інтелектуального опрацювання документів на основі NER», керівник проєкту Болдак Андрій Олександрович, к. т. н., доц. каф. ОТ ФІОТ, затверджені наказом по університету від «07» травня 2020р. № 1081-с

2. Термін подання студентом проєкту _____

3. Вихідні дані до проєкту: технічна документація, теоретичні дані, інтернет-публікації за темою роботи

4. Зміст пояснювальної записки: проведення аналізу предметної області та існуючих передумов для розробки, аналіз аналогів рішень та допоміжних модулів для виконання роботи, проведення тестування розробленого сервісу та аналіз методів роботи з отриманими результатами.

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо)

6. Консультанти розділів проєкту*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Сімоненко В.П., проф.		

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1	Затвердження теми роботи	01.09.2019	
2	Вивчення та аналіз завдання, огляд літератури по темі	25.12.2019	
3	Аналіз аналогів рішень та допоміжних модулів для виконання роботи	01.02.2020	
4	Проектування та розробка сервісу для опрацювання документів на основі розпізнавання іменованих сутностей	05.03.2020	
5	Проведення тестування розробленого сервісу та аналіз методів роботи з отриманими результатами	01.04.2020	
6	Оформлення матеріалів роботи	17.05.2020	
7	Передзахист	26.05.2020	
8	Захист	…	

Студент Вадим ЛЕВКІВСЬКИЙ

Керівник Андрій БОЛДАК

Анотація

В даному дипломному проєкті проведено аналіз задач пошуку інформацій та розпізнавання іменованих сутностей. Розглянуто аналоги та методи рішення задач NLP. Модифіковано сервіс розпізнавання іменованих сутностей для української мови під специфіку задачі даного дипломного проекту. Розроблено методи оптимального зберігання результатів розпізнавання іменованих сутностей та подальшого використання отриманих даних. Розроблено методи та алгоритми побудови семантичних мереж сутностей. Проведено аналіз різних видів баз даних для зберігання та візуалізації даної мережі. Розроблено методи адміністрування графічної баз даних сутностей. Приведено приклади роботи з семантичною мережею іменованих сутностей.

Annotation

In this Bachelor's work the analysis of problems of search of the information and recognition of the named essences is carried out. Analogues and methods of solving NLP problems are considered. The service of recognizing named entities for the Ukrainian language has been modified according to the specifics of the task of this diploma project. Methods of optimal storage of the results of named entities recognition and further use of the obtained data are developed. Methods and algorithms for constructing semantic networks of entities have been developed. The analysis of different types of databases for storage and visualization of this network is carried out. Methods of administration of graphic databases of entities are developed. Examples of work with a semantic network of named entities are given.

ВІДОМІСТЬ ДИПЛОМНОГО ПРОЄКТУ

№ з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
1	А4		Завдання на дипломний проєкт	2	
2	А4	ДП 6115. 00.000 ВП	Відомість дипломного проєкту	1	
3	А4	ДП 6115. 01.000 ТЗ	Технічне завдання	3	
4	А4	ДП 6115. 02.000 ПЗ	Пояснювальна записка	62	
5	А3	ДП 6115. 03.000 Д1	Структура модулів сервісу. Схема структурна	1	
6	А3	ДП 6115. 04.000 Д2	Послідовність передачі повідомлень для завантаження документу. Схема функціональна.	1	
7	А3	ДП 6115. 05.000 Д3	Опрацювання запиту обробки документа. Схема принципова	1	

				ДП 6115 00.000.00 ВП		
	ПІБ	Підп.	Дата			
Розробн.	Левківський В.В.			Відомість дипломного проєкту	Лист	Листів
Керівн.	Болдак А.О.				1	1
Консульт.					КПІ ім. Ігоря Сікорського Каф. ОТ Гр. ІО-61	
Н/контр.	Сімоненко В.П.					
Зав.каф.	Стіренко С.Г.					

ТЕХНІЧНЕ ЗАВДАННЯ

до дипломного проєкту

освітньо-кваліфікаційного рівня бакалавр

на тему: “ Веб-сервіс для інтелектуального опрацювання документів на основі NER ”

Київ – 2020 року

ЗМІСТ

1. НАЙМЕНУВАННЯ ТА ОБЛАСТЬ ЗАСТОСУВАННЯ

Дане технічне завдання розповсюджується на розробку програми інтелектуального опрацювання документів за допомогою розпізнавання іменованих сутностей.

Область застосування: веб сервіси для обробки документів в яких присутня база даних для зберігання.

2. ПІДСТАВИ ДЛЯ РОЗРОБКИ

Підставою для розробки служить завдання розробки програми побудови семантичної мережі за допомогою NER та методів подальшої роботи з нею.

3. МЕТА ТА ПРИЗНАЧЕННЯ РОЗРОБКИ

Метою даного проекту є розробка сервісу інтелектуального опрацювання документів, що реалізує виконання основних запитів аналізу сутностей та зв’язків в тексті.

4. ДЖЕРЕЛА РОЗРОБКИ

Джерелами для розробки служать науково-технічна література з комп’ютерних технологій, публікації в періодичних виданнях, наявні сервіси та бібліотеки для розв’язання задачі NER, перелік основних задач електронного документообігу.

5. ТЕХНІЧНІ ВИМОГИ

5.1. Вимоги до розроблюваного продукту

- Розробка інтерфейсу для графічного вводу графів алгоритмів
- Виконання розбиття графу алгоритму на яруси.
- Виконання програмної емуляції роботи алгоритму адаптивної реконфігурації

- Розробка засобів візуалізації результатів моделювання.

5.2. Вимоги до програмного забезпечення

- Графічна база даних Neo4j версій 3.x та 4.x
- MongoDB

5.3. Вимоги до апаратного забезпечення

- Комп’ютер на базі процесору Intel Pentium 5 і вище
- Оперативної пам’яті не менше 2 Гбайт
- Наявність серверу для зберігання інформації (опціонально)

Пояснювальна записка
до дипломного проєкту
на тему: «Веб-сервіс для інтелектуального опрацювання документів на основі NER»

Київ – 2020 року

Зміст

ВСТУП

Зараз все більше організацій та звичайних користувачів переходить на електронний документообіг. При цьому важливо, щоб системи документообігу мали функціонал для опрацювання, класифікації та швидкого пошуку необхідних документів або даних, що містять в документах. Функція пошуку має забезпечувати гнучкий пошук по словам запиту, з виводом синонімічних до запиту даних або інформації, що може бути корисна. Також корисною задачею в електронних архівах є виявлення ключових даних та об'єктів (персон, назв установ, дат, місць, чисел, посад людей), зв'язків між ними та пошук прихованих знань. З вирішенням цих завдань може допомогти задача розпізнавання іменованих сутностей (Named-entity recognition) з побудовою зв'язків між ними. Результати запитів користувачів мають бути візуалізовані для кращого їх сприйняття користувачем. Для цього варто забезпечити можливість виводу результату у вигляді графічних структур, наприклад графіку.

Отримане рішення буде корисно не тільки для архівів електронного документообігу, а для аналізу інформаційних структур, що містять текст. Наприклад веб сторінок, блогів, сервісів новин. Задача NER може не тільки знаходити слова, що є іменованими сутностями, а й наповнювати їх семантичними характеристиками, за допомогою інших сутностей та слів, що знаходяться в тексті.

РОЗДІЛ 1 АНАЛІЗ ЗМІСТУ ПОСТАВЛЕНОЇ ЗАДАЧІ ТА МЕТОДІВ РІШЕННЯ

1.1 Поняття документообігу

Документообіг - рух документів в організації з моменту їх отримання чи утворення до завершення виконання чи відправки.

Вірна організація документообігу сприяє оперативному проходженню документів в організації управління, пропорційному завантаженню підрозділів, відділів та посадових осіб, що показує позитивний вплив на процес управління в цілому.

Виділяють такі види документообігу: централізований документообіг (вся документація централізовано реєструється); децентралізований документообіг (реєстрація документів у кількох місцях за умови річного документообігу 100 тисяч і більше документів, а також за наявності територіально уособлених структурних підрозділів та певних особливих умов роботи); змішаний документообіг (найбільш важлива внутрішня документація та листування керівництва реєструється у канцелярії, решта документів – у структурних підрозділах). [1]

Електронний документообіг (обіг електронних документів) - сукупність процесів створення, опрацювання, відправлення, передавання, одержання, зберігання, використання та знищення електронних документів, що виконується із застосуванням перевірки цілісності та у разі необхідності з підтвердженням факту одержання таких документів. Відповідно до Закону України «Про електронні документи та електронний документообіг» електронним документом визнається документ, засвідчений електронним цифровим підписом. А документ в електронній формі – документ, інформація в якому зафіксована у вигляді електронних даних без електронного цифрового підпису (в сканованій формі). Електронний підпис (ЕП) - дані в електронній формі, які додаються до інших електронних даних або логічно з ними пов'язані та призначені для ідентифікації підписувача цих даних. ЕП є обов’язковим реквізитом електронного документа,

який використовується для ідентифікації автора та (або) підписувача електронного документа іншими суб'єктами електронного документообігу. Відносини, пов'язані з використанням електронних цифрових підписів врегульовано Законом України «Про електронний цифровий підпис». Поступовість впровадження електронного документообігу. [1]

Електронний документообіг дозволяє підприємству отримати наступні переваги:

- одноразова реєстрація документа, що дозволяє безпомилково ідентифікувати його в системі;
- скорочення матеріальних витрат (папір, фарба, площі для зберігання)
- паралельне виконання операцій обробки докуметів, що скорочує час руху документа і підвищує оперативність виконання;
- безперервний рух документа, що дає можливість виявити відповідального за його виконання в будь-який момент процесу;
- виключає можливість дублювання через єдину база документів;
- результативний пошук документа за наявності про нього мінімальної інформації;
- ефективна система звітності, що дозволяє контролювати рух документа на кожному етапі документообігу.[2]

Всі системи електронного документообігу можуть бути класифіковані за кількома ознаками:

- СЕД з розвиненими системами зберігання і пошуку інформації. Їх друга назва - електронні архіви.
- СЕД з розвиненими системами маршрутизації, що забезпечують рух документів по заданих маршрутах.
- СЕД з системою підтримки управління організацією та накопичення знань. Зазвичай ці системи поєднують в собі властивості двох попередніх.

При цьому в такій системі можливе використання як жорсткої, так і вільної маршрутизації. Подібні СЕД використовуються в великих компаніях і державних структурах.

- СЕД з підтримкою спільної роботи співробітників. Такі системи націлені на організацію колективної роботи співробітників навіть у тому випадку, якщо вони розділені територіально. Надають можливість пошуку інформації, обговорень та призначення зустрічей, включаючи реальні і віртуальні, а також сервіси зберігання і публікації документів.
- СЕД з додатковими сервісами: управління проектами, електронна пошта, білінг, сервіс CRM.

Найбільш затребуваними функціями СЕД є:

- Зі зберігання й пошуку документів.
- Підтримка діловодства.
- Маршрутизація і контроль виконання документів: складання маршрутів документів, підтримка дій під час маршрутів, повідомлення співробітників про вступ нового документа, автоматичний контроль термінів виконання.
- Складання аналітичних звітів, таких як звіт про поточну зайнятості, про виконання робіт по документам і про прострочених дорученнях.
- Забезпечення інформаційної безпеки, включаючи аутентифікацію користувачів, підтримку електронного цифрового підпису, шифрування документів і листів, аудит роботи в системі.

З самого початку заснування суб'єкт господарювання, в залежності від потреб, взаємодіє з різними суб'єктами - це і держава, і бізнес - партнери, і контрагенти, і звичайні споживачі. Для ідентифікації інформаційної та економічної взаємодії вже давно набули поширення терміни, які влучно та зрозуміло визначають тип взаємодії (Рис.1.1):

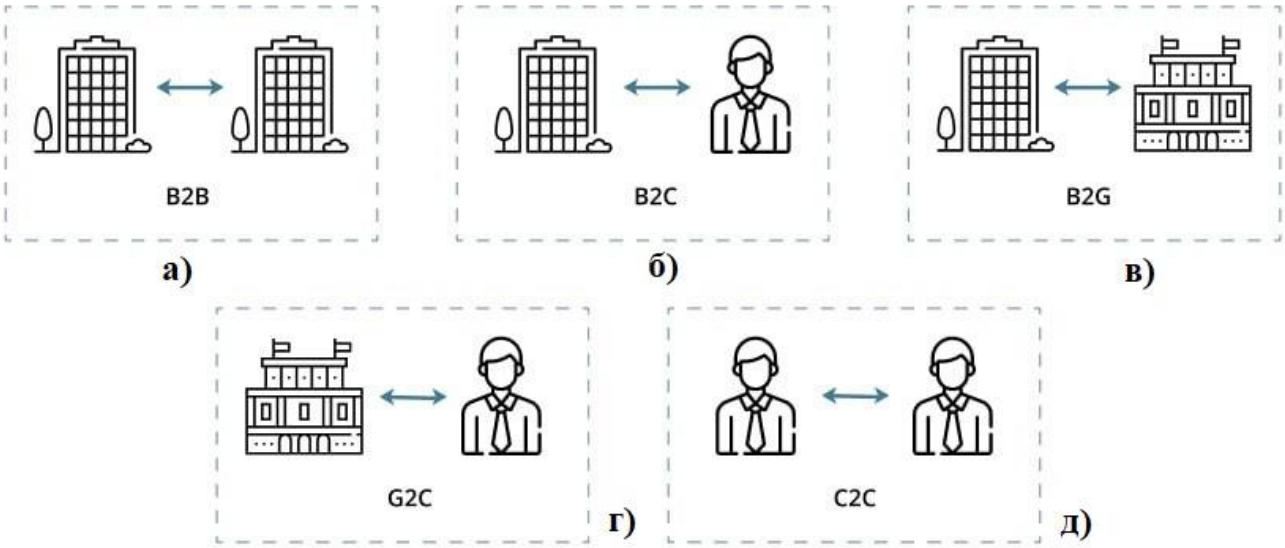


Рис. 1.1 Типи взаємодії: а) бізнес – бізнес; б) бізнес – держава; в) бізнес – споживач г) споживач – держава; д) споживач – споживач. [3]

Детальніше про кожен тип:

- B2B (англ. business to business) — бізнес - бізнес. Тобто, взаємовідносини між підприємствами. Для даного документообігу немає чіткого стандарту для всіх документів, бо різні підприємства мають різні внутрішні стандарти. Для забезпечення такого документообігу потрібні сервіси з базовою підтримкою редагування файлі. Для такого типу документообігу популярне використання електронних архіві з великою кількість функцій. Автогенерація документів не характерна;
- B2G (англ. business-to-government) — бізнес - держава. Відносини між бізнесом та державою. Характерний чіткий стандарт для всіх видів документів. Документи передаються рідко та переважно пакетами. Сервісами для підтримки документообігу є державні сайти. Характерне заповнення анкет та шаблонів для створення документів;
- B2C (англ. business-to-consumer) — бізнес - споживач. Термін означає комерційні взаємовідносини з фізособою - «кінцевим» споживачем. Дуже рідко зустрічається стандарти оформлення документів. Основні документи це файл з описом послуг, товарів, реклама. Серед документів юридичного

характеру: договір про послуги, гарантійні документи товарів, квитанцій та інші фінансові документи про оплату ;

- G2C (англ. government-to-consumer) — споживач - держава. Тобто, взаємовідносини між державою та фізособою - «кінцевим» споживачем. Всі документи стандартизовані формат яких регулюється законом та юридичними нормами. Для даного документообігу дуже корисними є функції: автоматичного заповнення документів, виловлення сутностей та ключових слів. Прикладами є державні сайти для звернення громадян. Характерна прив'язка великих баз даних для зберігання документів завантажених користувачами. ;
- C2C (англ. consumer-to-consumer) — споживач - споживач. Тобто, комерційні взаємовідносини фізособи з фізособою. Рідко зустрічається стандарти оформлення документів. Основні шаблони визначенні законами та їх люди вивчають в курсах вивчення мов: заяви, довіреності, лист. Великий сегмент підтримується через сервіси файлообмінники. Певні правила такого документообігу визначенні правилами, що формувались в суспільстві. Наприклад в початку документу вказувати свої дані.

1.2 Архів

Електронний архів — це система структурованого зберігання електронних документів, що забезпечує надійність зберігання, конфіденційність і розмежування прав доступу, відстеження історії використання документів, швидкий і зручний пошук. [4]

Інтенсивний розвиток інформаційно-телекомунікаційних технологій супроводжується стрімким накопиченням документів та інформаційних ресурсів з цифровими носіями. Вирішення цієї проблеми пов'язане із застосуванням електронних інформаційних технологій в організації діловодних процесів на підприємствах, в організаціях, установах, органах державної влади, органах

місцевого самоврядування тощо. При цьому для кожного типу структури присутня своя специфіка організації обліку, обробки, зберігання та вилучення документів. Серед новітніх інформаційних технологій установи, підприємства велику роль починає відігравати електронний архів. Це викликано перевагою деяких ознак електроної докумнетації перед паперовою. Діловодна та архівна практика засвідчує, що заміна ручних технологій на електронні, дозволяє серйозно прискорити документообіг, управління інформацією та її використання.

Метою впровадження електронного архіву в діяльність установі є: створення повного централізованого архіву документів з можливістю динамічного управління доступом до інформації; введення єдиної технології роботи з електронними документами в межах установи, що забезпечує захищеність і керованість електронним архівом; організація робочих місць відповідно до сучасних високоефективних технологій колективного використання і роботи з документами; забезпечення оперативного доступу до текстових та графічних образів документів у межах вказаних прав за допомогою повнотекстового пошуку та пошуку за атрибутами документа; визначення місця фізичного знаходження оригіналу документа в архівах паперових документів або у виконавця як результат пошуку в електронному архіві; створення добірок документів, що відповідають заданим користувачем критеріям відбору; надання можливості користувачу працювати як з переліком документів, так і з окремими документами або їх фрагментами.[5]

Слід зауважити, що документообіг все більше переходить масивні електроні архіви, що можуть нараховувати да сотні тисяч документів та файлів. Організація електроного архіву передбачає: базу для зберігання документів, методи завантаження та перевірки документів на відповідність правилам ведення документації у певній установі, режими доступу для користувачів, методи обробки документів їх перетворення, ідентифікації документів та побудові атрибутів документу, функцій для адміністрування. Серед важливих переваг електронних архівів є компактне зберігання документів за рахунок оптимізації вмісту та

можливості створювати необмежену кількість копій для паралельної роботи декількох користувачів. При інтеграції електронного документообігу в установу важливу роль відіграє визначення правил роботи з документами, їх передачі між відділами, узгодження, та стандарту ведення документації. Технології електронного підпису забезпечує швидке узгодження та погодження документації.

Зазвичай стандарти документообігу регулюються законами та нормами країни, і правилами визначенні в самій установі. Електронний архів має передбачати можливість швидкого пошуку документів за вмістом, атрибутами та іншими критеріями.

1.3 Пошук та його різновиди

Пошукові системи – системи, що забезпечують пошук необхідної інформації в архівах даних. Для пошуку користувач формує пошуковий запит. Дані аналізуються і створюється список об’єктів, що підходять під запит. Список таких об’єктів сортується по ступеню відповідності пошуковому запиту і виводиться, як результат пошуку. В ролі об’єктів пошуку можуть виступати: сайти, відео, документи. Причому в ролі пошукового запиту може виступати: текстовий рядок, набір фільтрів, зображення, аудіо- або відео-файл. Наприклад, сервіси Find Face (Рис. 1.2), Search4faces дозволяють знаходити по фотографії облікові записи людей в соціальних мережах, а сервіс Pipl дозволяє ефективно знаходити інформацію про конкретну людину. FindSounds знаходить конкретні звуки у відкритих ресурсах. Через великі обсяги інформації, що створюються кожного дня в конкретиці в інтернет мережі, все більше інтернет сервісів мають функціонал пошуку для швидкого отримання необхідного користувачеві вмісту.

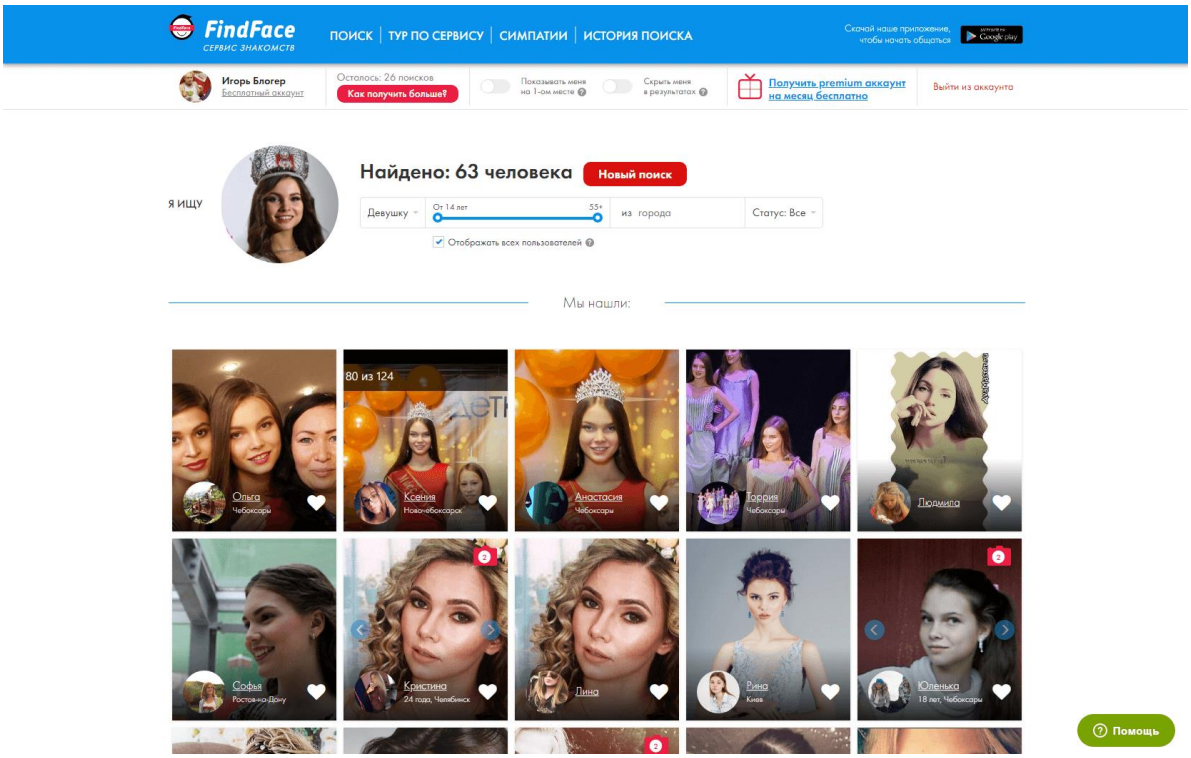


Рис. 1.2 Результати запиту в сервісі Find Face.

Лідером пошукових систем є Google, який дозволяє шукати веб-сторінки, зображення, відео та місця на карті. Пошук реалізовується по текстовим запитам і вбудованим функціям, схожим зображення, голосовим запитам. Також є додаткові сервіси, що забезпечують пошук та аналіз даних, такі як:

- `Google Trends – для відслідковування та аналізу трендів в суспільстві по зміні частоти пошукових запитів.
- Google News – агрегатор новин, що забезпечує пошук найактивніших новин.
- Google Scholar – сервіс для пошуку інформації по галузі знань та статей по ключовим словам.
- Google Patents Search – пошук по патентам.

1.3.1. Пошукові каталоги

Інформація зберігається у вигляді тем - категорій і підкатегорій. Переваги каталогів в якості матеріалу, який являє собою класичну і найбільш популярну інформацію за поданою темою. Тому, каталоги є першоджерелами для

ознайомлення з новими темами, які незнайомі користувачеві. Однак, маловідомі ресурси, і при цьому досить прогресивні, в таких каталогах зустрічаються рідко.

Наприклад, до пошукових каталогів можна віднести всім відомий сайт wikipedia.org, що містить довідкову інформацію про всіх і про все.

Зазвичай пошукові каталоги мають ієрархічну структуру. Все починається з розподілу по галузям знань, потім категоріям певної галузі, категоріям і т.д. На найнижчому рівні знаходяться посилання не ресурси з коротким описом. В середині конкретних часто підтримується сортування по алфавіту, даті додавання та індексу цитування. В деякому розумінні, перші пошукові каталоги не є повноцінними пошуковими системами, бо пошук обмежувався вмістом каталога, а не мережою Інтернет.

З початку існування розробкою каталогів займались експерти, що класифікували сайти, сторінки та документи, і розміщували їх всередині ієрархічної структури. Найбільший каталог мережі DMOZ (Open Directory Project) об'єднує інформацію про 5 мільйонів ресурсів, тоді як база пошукової системи Google нараховує більш ніж 8 мільярдів документів.[6]

Прикладом пошукового каталога є сервіс Yahoo!, здебільшого на початкових етап свого існування. Він є базою URL-адрес сайтів, класифікованих по різній тематиці. Пошук здійснюється починаючи з вказання теми, що вас цікавить, і конкретизації за допомогою підсказок каталогу. Також забезпечувався пошук по ключовим словам. Користувач вказує через пробіл слова, що мають зустрічатись в пошуковій інформації. Система саме підбирає посилання на необхідну вам дані.

1.3.2. Рейтингові системи

Варіація пошукового каталогу, яка передбачає організацію видачі за кількістю звернень відвідувачів. Тобто, основним критерієм є популярність ресурсу, яка, на жаль, не завжди свідчить про його корисності, змістовності та

інформативною цінності. Тому, рейтингові системи найбільше підходять для пошуку розважальних і новинних матеріалів. Такий метод відноситься до нелінійного пошуку. Результатом пошуку буде список сервісів по конкретній тематиці за популярністю, тобто переглядами сторінки.

Прикладом рейтингової системи є знаменитий портал оцінки популярності інтернет-ресурсів alexa.com. Сервіс видає інформацію про кількість відвідувань, рейтинг, трафік та інше по конкретному інтернет ресурсу.

Rambler's Top100 – система веб аналітики та лічильник відвідувань сайтів. Основу сервісу становить створення рейтингів по конкретним покажчикам, як: час перебування на сайті, характеристики користувачів, пристрою та операційна система з яких відвідується сайт, джерела звідки здійснювався перехід. [7]

1.3.3. Пошукові покажчики

Окремий клас, який виділяє дані пошукові системи серед всіх інших, які організовують пошук інтернет-ресурсів за ключовими словами. Успіх пошуку повністю залежить від слів, заданих в запиті, і пошук з використанням ключових слів і фраз далеко не завжди буває ефективним, зважаючи на різноманітність мови. Однак, коли необхідний рідкісний матеріал на конкретну тему, правильно підібрані ключові слова роблять даний вид пошукової системи дуже ефективним. Основу становлять допоміжні індекси по яким користувач може швидко знайти необхідні йому об'єкти. Цей вид характеризує більшу частину пошукових систем. Прикладом пошукових покажчиків є списки термінів, сутностей в кінці книги. Там перераховані всі важливі терміни книги з вказанням сторінок на яких вони зустрічаються. На деяких веб ресурсах можна знайти панелі де перераховані ключові категорії об'єктів сервісу. Наприклад сервіси новин де зазвичай на верхній панелі знаходяться посилання на категорії новин, зустрічаються панелі з блоками найбільш популярних новин. Деякі сервіси забезпечують автоматичне формування

пошукових покажчиків по рейтингу найбільш популярних серед користувачів сторінок [8].

1.3.4. Атрибутивний пошук

Пошук який заснований на пошуку об’єктів по їх конкретним характеристикам. Наприклад: автори творів, тип об’єкта, ціна, колір, вартість, компанія виробник. В ролі значення атрибутів виступають їх якісні та кількісні показники (зелений, дорогий, зроблений в Китаї). Також включає можливість використання логічних функцій для пошуку по групі характеристик з поєднанням їх логічними функціями. Користувач вказує список атрибутів та параметрів атрибутів, що його цікавлять або класифікують пошукові об’єкти. Атрибути об’єктів порівнюються з атрибутами пошукового запиту. Результатом пошуку виступає група об’єктів, які мають визначенні користувачем характеристики. Простим прикладом інтернет магазин де ви вказує необхідні вам характеристики товару і сайт видає вам список товарів, що вам підходять. Даний вид пошуку історично закріпився за архівами бібліотек. Ви обираєте початкову літеру назви книги або автора, вид літератури.

Атрибутивний пошук є доволі простий і швидким у реалізації. Дозволяє отримувати точний, а не ймовірнісний результат. Не потребує розробки складних пошукових механізмів та великих атрибутивних описів об’єктів.

1.3.5. Повнотекстовий пошук

Базується на проходженні всього тексту документу і порівняння його з пошуковий запитом. Кожне слово аналізується і перевіряється на відповідність слів запиту. У випадку великих баз даних, застосовується методика індексації. Індексація полягає в скануванні тексту всіх елементів бази і складання пошукових термінів (покажчиків). Потім відбувається порівняння запиту та пошукових

термінів. Даний вид пошуку дуже широко був поширений в 90-х роках. Прикладом є пошукова система AltaVista, що втратила свою популярність після появи Google. Наразі повнотекстовий пошук використовується для невеликих об’ємів даних із застосуванням стратегії «послідовного сканування». Повнотекстовий пошук дозволяє шукати і словоформи слів пошукового запиту. Тобто по пошуковому запиту «головний біль» буде знаходити результати, що містять «головному болю», «біль голови» і т.д. При цьому для більш швидкої роботи при скануванні текстів ігноруються так звані стоп-слова. Пошукові алгоритми спираються на іменники, бо вони часто складають основу мову, рідше прикметники, прислівники та дієслова. В силу специфіки алгоритмів пошуку є корисна методика використання рідкісних слів в запиті. Так користувач швидше отримає необхідний результат. Вказанні в запиті точних слів та термінів забезпечує збільшення якості пошуку та знаходження підходящих результатів. В такому виду пошуку рідко використовується методика пошуку також і по синонімам слів пошукового запиту. Про те при такій методиці спочатку алгоритм буде шукати по точних спів падінням і словоформам, потім по синонімам слів запиту.

1.3.6. Семантичний пошук

В текстів визначаються терміни і зв’язки між ними. Пошук відбувається по змістовному складу. Поділяється на повнотекстовий пошук по всьому документу з використанням попередньо визначеними індексами та пошук по метаданих, де пошук відбувається по атрибутах, що описують об’єкти підтримувані системою, такі як автор, дата, посада людини. Даний пошук орієнтується на зміст, тобто атрибути і значення, що знаходяться за текстом даних. Причому семантику не потрібно визначати у всьому тексту, а тільки в ключових типах сутностей. Переважно у всіх даних доступних для сервісу пошуку проводиться семантичний аналіз для визначення змістовного складу та створення атрибутів, що будуть до нього приєднанні. Після чого сервіс може використовувати ці дані, як об’єкти серед яких проводиться пошук. Проміжним етапом між пошуковим запитом користувача

і результатом-відповіді сервісу на запит, є процес перетворення запиту на людській мові в специфічну машину мову запитів алгоритму семантичного пошуку.

Наразі семантичний пошук все ширше починає використовуватись. Різні інтернет сервісу додають його для реалізації конкретних пошукових задач, або здач аналізу змісту. Наприклад у 2012 пошукова система Google додали семантичні алгоритми пошуку до свого функціоналу. Це збільшило релевантність результатів пошуку. У сервісу з'явились нові функції результату:

- тепер сервіс міг дати точну відповідь на запит «топ 20 найбільш забудованих міст світу», в не список більш-менш підходящих посилань;
- на запит про конкретного автора, користувач в результаті отримував посилання на пов'язаних з цим автором письменників та інформацію де можна придбати книги.

Семантичний пошук забезпечує розв'язок задачі пошуку прихованих неявних знань. Для цього переважно використовують структури, що складаються із знайдених в масивах даних семантичних об'єктів та зв'язків між ними.

1.4. Класифікація задач Natural Language Processing

1.4.1 Добування даних

Добування даних (Data Mining) – процес автоматизованої обробки великих баз даних для отримання конкретної інформації або корисної знань в цілому. Основа задачі знаходження шаблонів, неявних зв'язків у тексті. Метою аналізу є виявлення правил, закономірностей, статистичних подій. В результаті опрацювання тексту неявні закономірності перетворюються в зрозумілі людині типу зв'язків. Поширеним використанням задачі добування даних: реферування даних та перетворення неструктурованих масивів інформації в синтаксично зв'язаний текст. Початково здача формулювалась, що у нас є велика база або архів

даних, в якому можуть знаходитись «приховані знання». Під прихованими знаннями розуміли знання:

- які не можна одразу виявити, тобто для їх виявлення недостатньо простого перегляду або інших стандартних методів;
- є новими, раніше не вивченими або не підтверджують виявлені раніше гіпотези;
- мають прикладну користь: становлять інформаційний інтерес або можуть дати економічну вигоду;
- є доступними для подальшої передачі, тобто можуть поширюватись за допомогою звичних методів інтерпритації та ілюстрації даних.

Data Mining вирішує задачі: кластеризація, прогнозування, візуалізація, підведення підсумків, класифікація, аналіз і виявлення відхилень, асоціація, оцінювання, аналіз зв'язків. [9] Детальніше про кожну задачу:

Класифікація (Classification) - рішення задачі класифікації визначає ознаки, за яким можна групувати об’єкт. В результаті групування об’єкти можна відносити до різних визначених наперед класам. Завдання класифікації можна вирішити використовуючи методи: найближчого сусіда (Nearest Neighbor); k-найближчого сусіда (k-Nearest Neighbor); байєсовські мережі (Bayesian Networks); k-найближчого сусіда (k-Nearest Neighbor); індукція дерев рішень; нейронні мережі (neural networks).

Кластеризація (Clustering) - логічним продовженням ідеї класифікації. В результати об’єкти розбиваються на групи. Класи не є визначеними наперед, тому процес вирішення задачі кластеризації є створення структури класів об’єктів за характеристиками цих об’єктів.

Асоціація (Associations) – виявлення закономірності між об’єктами або подіями в тексті за допомогою правил асоціації. Саме тут постає питання знаходження одразу неявних для людини зв’язків та закономірностей.

Пошук закономірностей здійснюється між кількома подіями, які відбуваються одночасно, що становить відмінність задачі асоціації від класифікації та кластеризації. Зазвичай використовують алгоритм Apriori для виявлення асоціативних правил.

Послідовність (Sequence), або послідовна асоціація (sequential association)

Дозволяє знайти тимчасові закономірності між транзакціями. Завдання подібне асоціації, але її метою є встановлення закономірностей між подіями, які відбуваються з деяким певним інтервалом у часі. Послідовність визначається високою ймовірністю ланцюжка пов'язаних у часі подій. Це завдання Data Mining також називають завданням знаходження послідовних шаблонів (sequential pattern).

Правило послідовності: після події А через певний проміжок часу відбудеться подія В.

Приклад. Після придбання квартири мешканці в 60% випадків протягом двох тижнів купують холодильник, а протягом двох місяців в 50% випадків купується телевізор. Рішення даної задачі широко застосовується в маркетингу та менеджменті, прикладом є управління циклом роботи з клієнтом (Customer Lifecycle Management).

Прогнозування (Forecasting). Задача дозволяє створити припущення та оцінку показників у майбутньому, аналізуючи особливості історичних даних. Для вирішення таких завдань застосовують нейронні мережі, методи математичної статистики, теорію ймовірності.

Визначення відхилень або викидів (Deviation Detection), аналіз відхилень або викидів. Рішення даної задачі дозволяє виявляти данні, що мають найбільше відхилення від множини звичайних даних. Тобто в процесі рішення ми визначаємо найбільш не характерні шаблони.

Аналіз зв'язків (Link Analysis) - задача знаходження залежностей в наборі даних та встановлення їхнього типу і виду зв’язку.

Візуалізація (Visualization, Graph Mining) – створення графічного образу аналізованих даних. В процесі вирішення завдання візуалізації використовуються графічні методи, що будуть наочну картину зв’язків об’єктів. Результатом можуть виступають діаграми, 2d і 3d об'єкти. Поширеними є також графи, де вершинами виступають об’єкти, а дугами типи зв’язків між ними.

Початково для роботи з даними та їх обробки користувались створеними спеціально мовами запитів для роботи з базою даних. Але згодом основу почала становити задача аналітики, для вирішення якої потрібні були математичні алгоритми в об’єднанні з методами навчання. До методів розв’язання відноситься: нейроні мережі, дерево рішень, еволюційні алгоритми.

1.4.2 Інформаційний пошук

Інформаційний пошук – задача пошуку інформації в неструктурованих базах даних. Об’єктами в базі можуть виступати текстові документи, зображення, відео. Суть задачі знаходження відповідних пошуковому запиту об’єктів. До завдань інформаційного пошуку відносяться: пошук інформації в документах, знаходження самих документів, вилучення метаданих та тегів з документів, пошук частин тексту, зображень, відео і звуку у локальних реляційних, гіпертекстових та графічних базах даних. Широкого застосування задача інформаційного пошуку набула в системах з великими базами даних для пошуку всіх об’ємів даних де згадується об’єкт, присутня інформація про подія або факт. Методи вирішення даної задачі можуть включати інтеграцію з іншими задачами NLP, спеціальна мова запитів, попереднє структурування масиву даних для легшого пошуку.

Орієнтуючись на ступінь залучення до інформаційного пошуку технічних засобів і участі в ньому людини розрізняють: "автоматизований", "машинний" і "ручний" інформаційний пошук.

В автоматизованих інформаційних системах задача інформаційного пошуку покладена на алгоритми та забезпечується і здійснюється із залученням

інформаційних, лінгвістичних, програмно-технічних, організаційних методів та засобів, і складених з них об’єднань. Можуть застосовуватись специфічні алгоритми на основі навчання нейронних мереж.

Головними ознаками якості результатів інформаційного пошуку є повнота, точність, релевантність та оперативність пошуку. Як критерій інформаційного пошуку, так і його результати є невизначеними. Цими ознаками інформаційний пошук відрізняється від ‘пошуку даних’.

Основний нюанс даної задачі, що в результаті у відповідь на пошуковий запит, система має видати інформацію, що становить найбільший пошуковий інтерес для користувача. При прямому пошуку, коли користувач отримує лише об’єкти з текстом, що місять чітку відповідність словам в пошуковому запиті, часто виникає обмеження, що результат матиме не весь масив об’єктів, що становлять пошуковий інтерес. Для цього корисним є системи, що шукають також по синонімічним тегам. Деякі системи використовують попередні запити даного користувача для того щоб краще розуміти, що саме треба знайти. Частою є практика коли пошукова система виводить також список запитів, які корелюють з запитом користувача.

1.4.3 Розпізнавання іменованих сутностей (Named-entity recognition)

NER – це задача знаходження в неструктурованому тексті іменованих сутностей та розпізнавання їх типу. У першій, класичній постановці, яка була сформульована на конференції MUC-6 в 1995 році іменовані сутності, це персони, локації і організації. З тих пір з'явилося кілька доступних корпусів, в кожному з яких свій набір іменованих сутностей. Зазвичай до персон, локацій і організацій додаються нові типи сутностей.[11] Тепер до іменованих сутностей відносять – імена, місця та території, дати, посади людей, назви компаній, грошові величини. Наприклад:

Після перемоги в морській битві біля Мегариди , Алексіос зустрічає 12 солдат на чолі зі спартанським царем Ніколаусом.

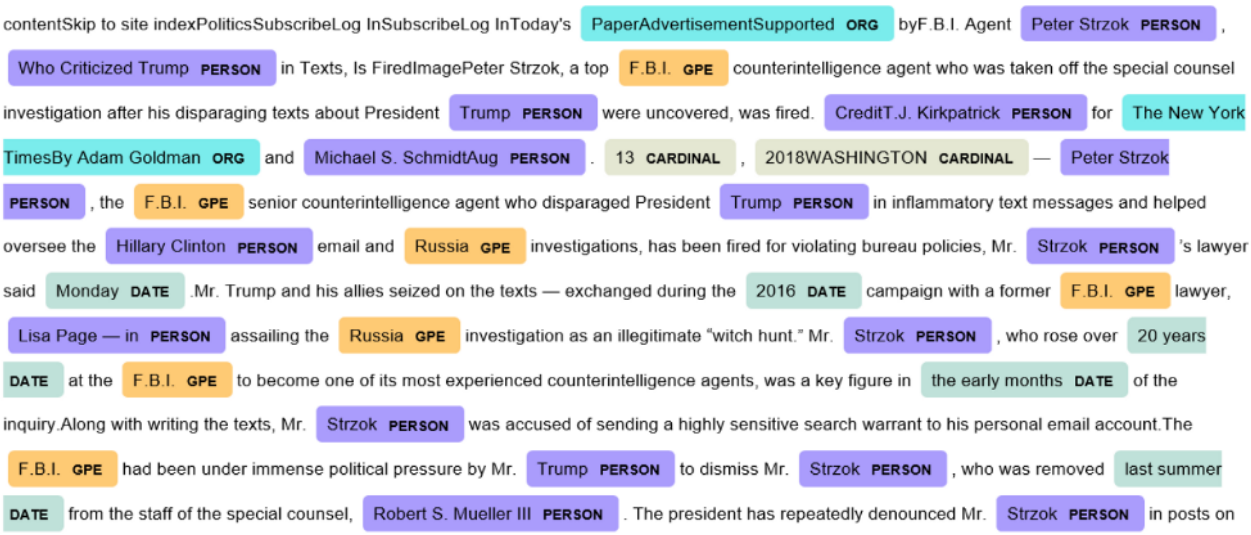
Після перемоги в морській битві біля [**Мегариди** | *місце*] , [**Алексіос** | *ім'я*] зустрічає [12 | *число*] [**солдат** | *посада*] на чолі зі [**спартанським царем** | *посада*] [**Ніколаусом** | *ім'я*].

За допомогою NER також можна вирішувати задачу розпізнавання частин мови, номерів телефонів, електронних адрес, різних типів даних. В результаті ми можемо виявити всі тексти, що містять згадування певної особи або місця. NER використовують для тегування тестів, тобто виявлення всіх тегів в тексті для опису об'єкту або його пошуку в базі за цими тегами. Це застосовується для товарів в інтернет магазинах, відгуків користувачів, електронних листів, дописів в блогах. Також NER дозволяє класифікувати відгуки користувачів про певний товар або компанію за емоційним характером, тобто виявляти позитивні та негативні коментарі. До основної підзадачі відноситься виявлення прихованих сутностей, у нашому прикладі посада визначена як «спартанський царь», а не просто «цар». Саме такі сутності як: президент перед ім'ям людини, район перед назвою території або область після – є прихованими і разом з основними, утворюють одну повну сутність.

В інтернеті задача NER набула широкого використання. Наприклад інтеграція задачі NER в пошук дозволяє більш точно підбирати об'єкти, що шукає користувач. Так в результаті по пошуковому запиту «Чорна машина німеччина» часто з'являються машини, що є іншого кольору, виготовленні не в Німеччині або мають опосередкований зв'язок, бо в тексті опису машини згадуються слова схожі на «Німеччина». Виділивши сутності в тексті і побудувавши мережу зв'язків, пошукова система зможе точно підбрати необхідний користувачеві набір даних.

Часто читаючи статті або дивлячись фільми в інтернеті, можна побачити, що на сайті також рекомендують схожі статті або фільми. Цей список часто будується на основі проаналізованих текстів інших сторінок сайту, виявлення сутностей і подальшого порівняння присутності схожих та однакових сутностей на різних

сторінках. Саме за допомогою таких алгоритмів ми можемо бачити підбірки подібних товарів в онлайн магазинах.



Рисинок 1.3 Приклад виявлення сутностей в тексті

Вирішувати задачу NER можна за допомогою словників сутностей. Слова в тексті порівнюють з вмістом словника у випадку співпадінь виділяється сутність і класифікується згідно визначеним словником груп. Даний метод у випадках великого об'єму словників вимагає великих затрат ресурсів. Іншим методом є визначення граматики та лексичних правил мови. Дані правила вписуються в якості алгоритмів і за їх допомогою проходить процес розпізнавання сутностей. Зазвичай програма просто пробує вгадати чи є дане слову\словосполучення сутністю і його тип. Різновидом методу є визначення специфічних правил характерних для сутностей даного типу тексту. До таких правил можуть відноситись морфологічні правила даної мови, бази даних, залежності між наповненням тестів пов'язаними сутностями. Для цим методів часто характерна проблема, що рішенням підходить тільки під конкретну мову. Наприклад якщо ми шукали сутності для російської мови і мали 80%, ти при запуску рішення для українського тексту ми отримаємо точність лише 30-40%. Для віддалений мов рішення взагалі не буде працювати для іншої мови. Найбільш популярним зараз методом вирішення задачі NER є нейроні мережі. Для цього потрібно навчити мережу за допомогою баз даних з правилами формування, шаблонів розпізнавання та словниками сутностей. Весь процес

рішення базується на принципі, що програма проходить текст і пробує вгадати чи є сутністю в ньому певне слово або послідовність слів. Відповідно чим більші бази даних були застосовані для навчання, тим більшу точність розпізнавання матиме нейрона мережа. Також для навчання можуть застосовуватись приклади вже опрацьованих текстів.

Багато мов програмування пропонують цілі модулі, що реалізують розпізнавання іменованих сутностей. Також існують цілі бібліотеки виявлення та роботи з іменованими сутностями, такі як: NLTK, HanLP, PullEnti, AdaptNLP, Spacy, NLTK, Stanford CoreNLP. За останні роки з’являються все нові веб-сервіси для вирішення задачі NER. В них присутня інтеграції з іншими задачами NLP, такими як: розпізнавання тексту на зображеннях, аналіз емоційного наповнення тексту, перетворення аудіо тексту, генерація тексту.

ВИСНОВКИ ДО РОЗДІЛУ 1

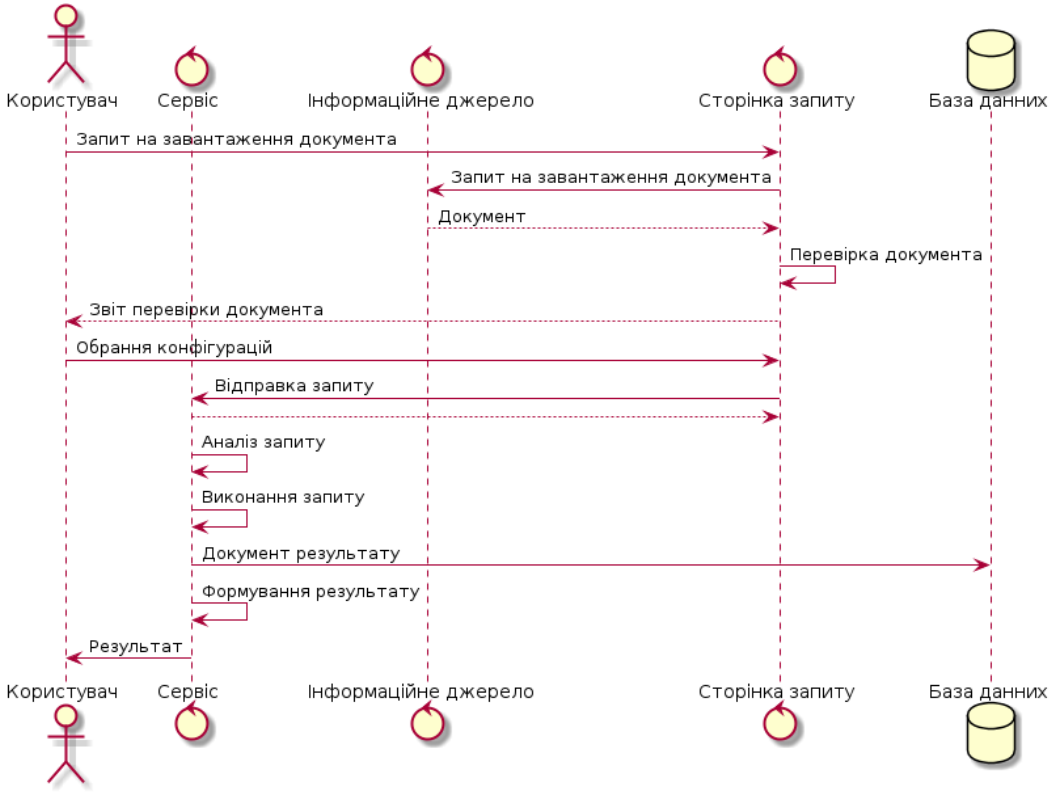
Електронний документообіг набуває все більшого поширення серед комерційний та державних установ. Важливою в електронному документообігу є наявність автоматизованої системи обліку, класифікації, опрацювання та швидкого пошуку документів. Для цього можуть застосовуватись рішення задач семантичного аналізу та виявлення іменованих сутностей. Наявність рішень пришвидшує внутрішні та зовнішні процеси організації, що мають зв’язок з документами. Це в свою чергу викликає зменшення витрати часу робітників, пришвидшення процесу прийняття рішень та зменшення грошових витрат.

Було проаналізовано види документообігу, їх властивості. Описано ключові характеристики основних методів пошуку. Описано задачі NLP, що можуть застосовуватись для інтелектуального опрацювання документів.

РОЗДІЛ 2 ОПИС ОПРАЦЮВАННЯ ЗАПИТІВ КОРИСТУВАЧА ТА ОБГРУНТУВАННЯ ОБРАНИХ МЕТОДІВ РІШЕННЯ

2.1 Життєвий цикл запиту на опрацювання документу

Весь цикл (рис. 2.1) починається із завантаженням документу в початкове вікно для запуску його подальшого опрацювання. Після користувач обирає необхідну йому конфігурації і запускає сервіс для опрацювання документу. Далі сервіс відповідно до встановлених користувачем конфігурації формую процес обробки, що складається з необхідних підпрограм та функцій, які відповідають за вказані конфігурації. Даний процес виконується і в результаті отримується опрацьований документ, що далі сервісом записується в базу даних. Зазвичай користувач має отримати результати опрацьованого документу так званий компонент відповіді. Іноді формат вже опрацьованого документу та компонента відповіді можуть різнитись. Тому в таких випадках починається процес корегування вже вписаного в базу даних документу під зручний для користувача результат.



Рисинок 2.1 Схема циклу запиту та опрацювання документу

Після отримання компоненту відповіді, користувач аналізує результат і приймає рішення про закінчення циклу опрацювання документу або повторного запуску документа на опрацювання але з новими конфігураціями. Більш детальніше про кожен етап циклу запиту опрацювання документу в наступних підрозділах. Будем порівнювати ці етапи для системи документообігу для типового виду компанії або установи, та додатку для якого характерні нетипові задачі по опрацюванню документа.

2.1.1 Завантаження документу та обрання конфігурацій обробки

Для сервісів типу додатків: користувач завантажує файл у тимчасове місце зберігання, наприклад веб-сервіс для опрацювання pdf-файлів ilovepdf (рис. 2.2). В ролі такого місця може виступати окремий розділ бази даних. Це характерно для сервісів з великою кількістю запитів. В таких сервісах формується черга опрацювання і щоб документ не втрачався, його записують в тимчасове місце зберігання до закінчення його закінчення формування компоненти відповіді користувачу.

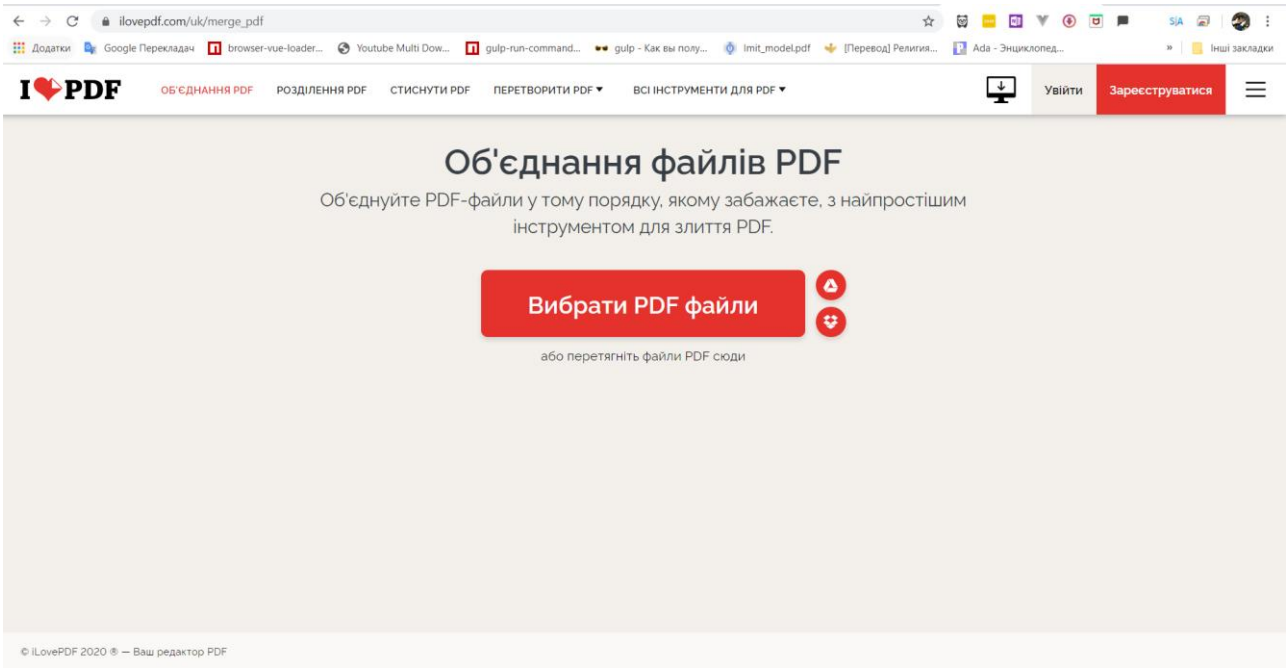


Рис.2.2 Панель сервісу для завантаження файлів [12]

Для сучасних сервісів характерна можливість завантаження документу з багатьох джерел, таких як: файлова система робочого комп'ютера користувача,

сховищ даних із сервера в хмарі, відкритих джерел по URL адресі. До них також входить і завантаження вже існуючих документів з бази даних, коли потрібно опрацювати старі документи.

На цьому етапі може здійснюватись перевірка файла на його неушкодження та відстуність шкідливого вмісту (рис. 2.3). За перевірку відповідають визначенні модулі сервісу. Після перевірки сервіс надсилає звіт перевірки документа. У випадку успішної перевірки звіт може містити коротке повідомлення або нічого. При цьому користувачеві стає доступна можливість обрати конфігурації опрацювання перевіреного документа. У випадку, якщо при перевірці сервіс виявив якісь ушкодження в документі або інші проблеми, звіт міститемо повідомлення про неуспішну перевірку і опціонально характер проблеми.

Потім користувач обирає необхідні конфігурації для опрацювання документу (рис. 7). До них відноситься: дії які потрібно виконати з документом, як відобразить результат виводу, що робити з опрацьованим документом, і т.д.

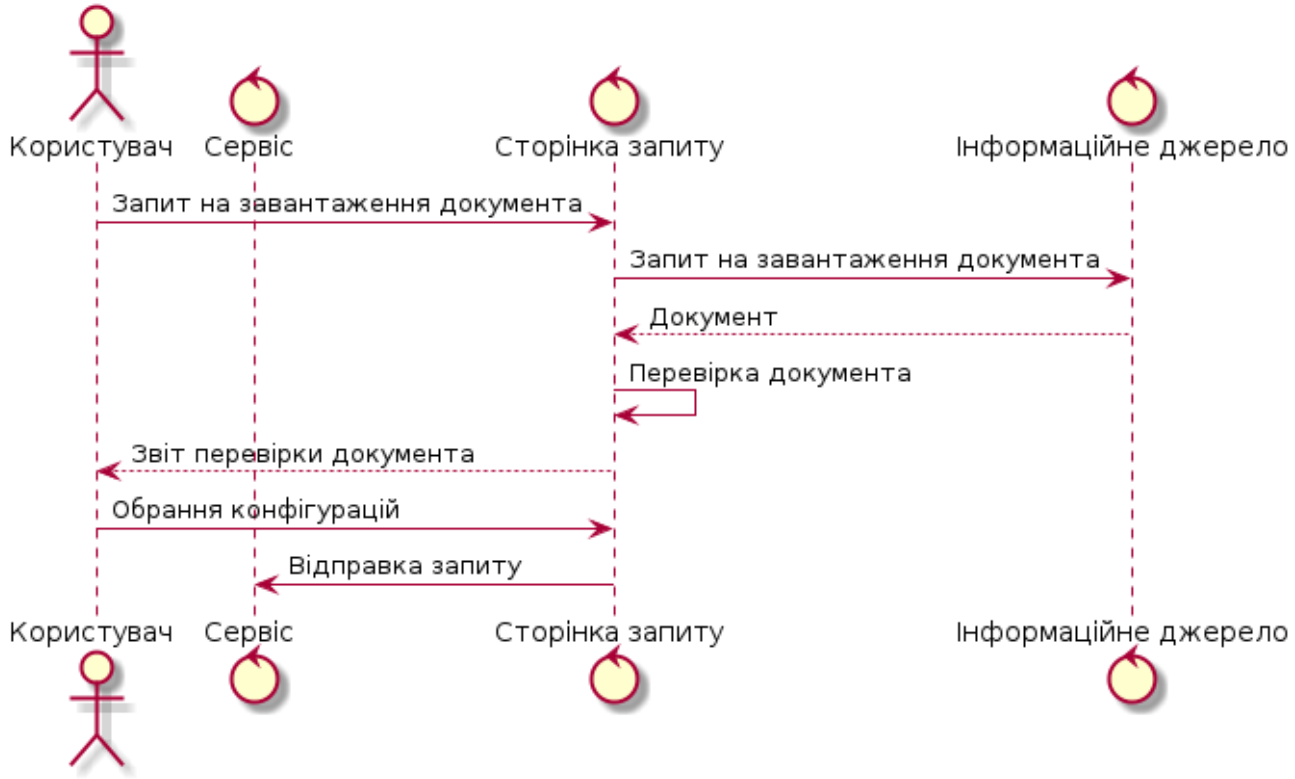


Рис. 2.3 Схема створення запиту користувачем

У діях які потрібно зробити, користувач обирає необхідні йому функції опрацювання та формат обробки (весь документ, тільки певні частини або елементи документу). У прикладі на рис. 2.4 користувач може обрати конфігурацію для: об’єднання документів в один або їх розділення на декілька файлів, перетворення у інший формат, зміни розміру файлу за рахунок стиснення, зашифрувати та редагування вмісту файлу.

Коли обрані всі необхідні конфігурації, користувач запускає сервіс для обробки документу.

У випадку систем документообігу на підприємствах та установах, документ розміщується з конфігураціями, вказаними користувачем в початковому сегменті обробки документації для електронного документообігу та передається у відділ вхідної документації для паперового або певного виду змішаного документообігу.

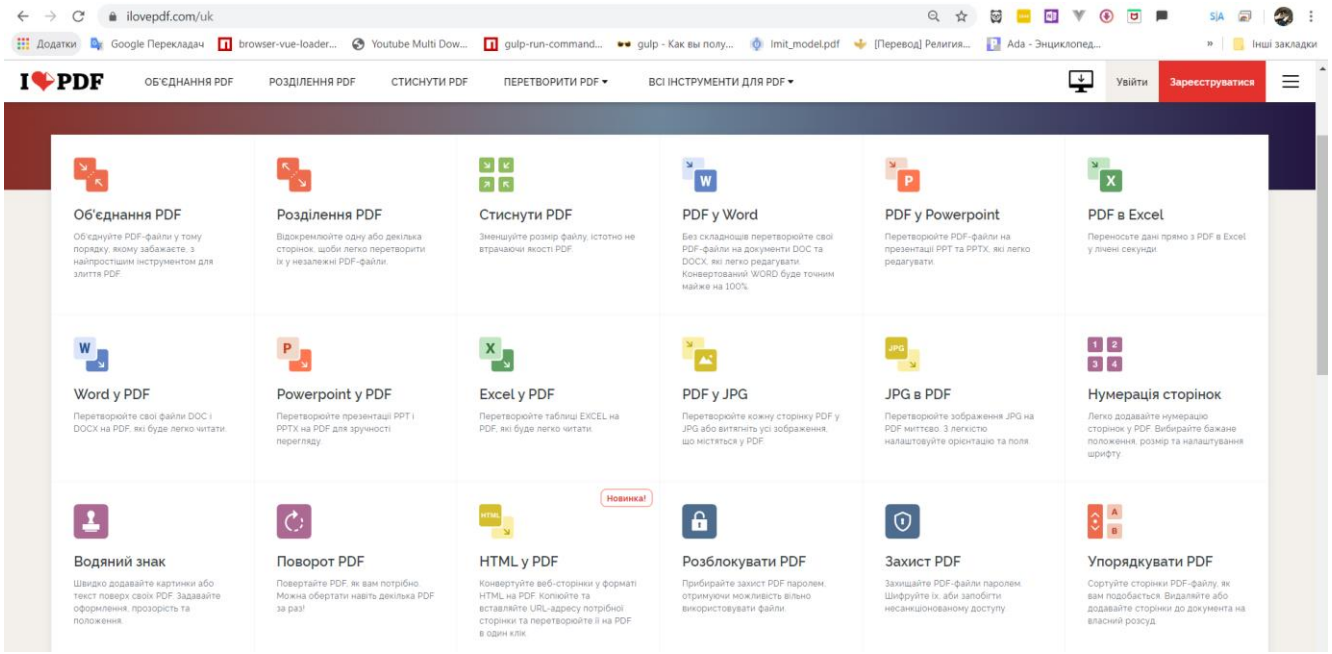


Рис. 2.4 Сторінка обрання конфігурацій [12]

В якості конфігурації виступають: який це вид документу (заява, розпорядження, проект рішення і т.д.) , якого рівня це є документ (всередині відділу, між віддільний або рівня компанії), яким відділам та департаментам варто передати його на опрацювання, типи інформації в документі (публічна, з обмеженим доступом, службова). Після відбувається аналіз відповідності

документу формату стандарту даного типу. Стандарт формату регулюються нормами та законами держави та визначеними правилами компанії або установи.

2.1.2 Створення та обробка компонента запиту

Створення компоненту запиту – це процес перетворення запиту користувача в завдання зрозуміле сервісу (рис. 2.5). Запит користувача аналізується системою та визначаються функціонал сервісу необхідний для вирішення завдань визначених конфігураціями. У випадку великого розміру та функціонального наповнення сервісу, доцільно формувати процес рішення що складатиметься тільки з програмних модулів, які необхідні для вирішення завдання. У такому випадку формуються оптимізовані процеси для запуску на сервері.

Наступним етапом є виконання компоненту запиту, тобто запуск та виконання відповідного процесу обробки. Веб-сервіси часто використовують стороні сервіси для процесу оброки запиту. В такому випадку характерні додаткові затрати часу, бо на якомусь етапі процесу потрібно отримати відповідь від стороннього сервісу.

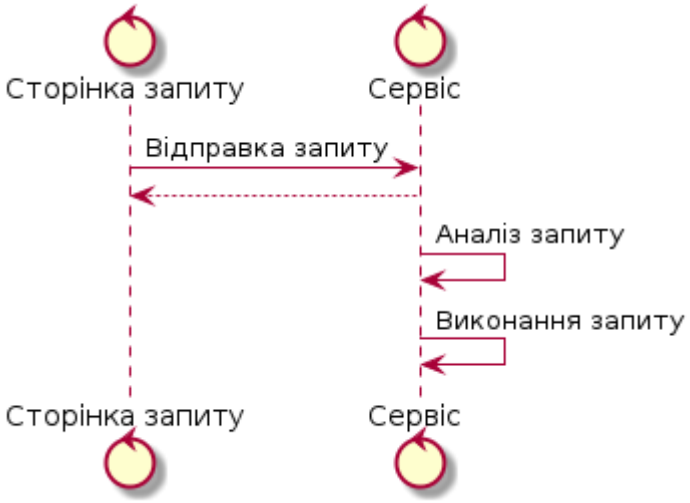


Рис. 2.5 Схема обробки запиту сервісом

Для роботи сервісів, де характерна висока кількість та інтенсивність запитів, потрібно вирішення задачі планування. Основними напрямки: досягнення швидкого виконання запиту і розподілення ресурсів під процеси обробки запитів користувачів.

Можливий додатковий етап, коли виникає збій: документ має певні проблемні ділянки, які сервіс не може опрацювати або існують підзадачі, що не передбаченні функціоналом сервісу. В таких випадках користувач отримує повідомлення від сервісу про проблеми у виконанні запиту. Також при високій завантаженості сервісу, доцільним є повідомлення користувача про те, що його запит поки не може виконатись і за можливості додається в чергу для майбутнього виконання.

2.1.3 Запис опрацьованого документу та вивід результату користувачу

По закінченню обробки, опрацьований документ записується в базу даних (рис. 2.6). Даний етап є опціональний і характерний тільки деяким видам сервісів: системи документообігу, баз даних, бази декларацій громадян. Після запису документа-результату в базу, його початкова версія видаляється.

Наступний етап вивід результату користувачу. Цей етап характерний для всіх сервісів. Результат залежить від формату виводу сервісу та можливих початкових конфігурацій вказаних користувачем. Наприклад для системи електронного документообігу, результатом може бути повідомлення про успішне виконання, з описом опрацьованого документу та виконаних дій. В таке повідомлення включається посилання для доступу для даного документу в базі даних. Для сервісу обробки користувацьких документів прикладу форматування, зміни вмісту, результатом буде опрацьований документ з можливостю завантаження. Відповідно, для таких сервісів характерна відсутність бази даних для зберігання опрацьованих файлів.

Рис. 2.6 Схема формування результату сервісом

Після отримання результату, користувач може прийняти рішення про повторне опрацювання документу. Доцільно в таких випадках, прописати в сервісі можливість використання попередньо опрацьованого документа в якості вхідного документа для нової обробки.

2.2 Запит та пошук семантичної інформації

Семантична інформація – це смисловий аспект інформації, тобто зв’язок між об’єктом, його суттю та словами, якими він виражений.

Будь-якому об'єкту графічного шару може бути поставлена у відповідність семантична інформація. Вказавши об'єкт на мапі, користувач може отримати семантичну інформацію, яка відповідає цьому об'єкту. І навпаки, задавши в запиті шукану комбінацію значень семантичних полів, користувач може дізнатися, яким графічним об'єктам вони відповідають. [13]

Семантична оцінка інформації характеризує її змістовність. Наразі постає проблема, що інформація на різних джерелах інформації: веб-ресурсах, форумах, архівах даних – представляється у вигляді наборів даних. В той же час для людини

звичне представлення інформації у вигляді зв’язків та асоціацій між об’єктами, даними. Формат інформації у вигляді взаємних залежностей більш зручний для запам’ятовування людиною. Пошукові сервіси зазвичай видають інформацію у вигляді підходящих масивів даних. Щоб опрацювати такі результати пошуку, людині потрібен час для визначення тієї інформації, що задовольнить її пошуковий інтерес. В процесі аналізу користувач будує семантичні зв’язки між даними. Проте, через те, що інформація може видаватись у вигляді частини статі, документу, характерне часткова або неправильна побудова семантичних зв’язків.

Напроти у системах пошуку, що підтримують семантичну аналіз даних при пошуку, користувачу більш зручно знаходити необхідну інформацію та будувати графи семантичних зв’язків. Такий підхід називають Semantic Web, що є набором стандартних технологій, що дають змогу створювати, описувати і використовувати онтологічні моделі. До технологій Semantic Web належать такі мови, як RDF (Resource Description Framework), RDFS (RDF Schema), OWL (Ontology Web Language) і SPARQL (Simple Protocol And RDF Query Language). [14]. Наразі технологія Semantic Web широко використовується для досліджень в галузях: систем управлінь знаннями, електронних бібліотек та пошукових систем.

Розглянемо як саме формується запит та реалізований пошук семантичної інформації. Будь-який пошуковий запит починається з формування користувачем пошукового терміну. Це може бути певний набір слів пов’язаних між собою, список термів, фільтрів для пошуку або їх об’єднання. Термами виступають слова, значення, що описують необхідній об’єкт (автор, дати публікації). Вказуючи пошуковий запит користувач також обирає конфігурації обробки інформації, формування результату та його вивід. Наприклад вказує, що варто шукати тільки в певного виду об’єкти (статті, пости, книги, текст певної мови, дати публікації, автора). Користувач старається за допомогою засобів мови в пошуковому запиті вказати, яка саме інформація йому потрібна.

Далі система аналізує дані і підбирає об’єкти, які підходять під запит. Порівнюється семантика запиту і семантику даних наявну в об’єктах для пошуку.

Часто системи вміють розпізнавати синоніми та об’єкти зі схожою семантикою. В такому випадку існують додаткові можливості для пошуку більшої кількості об’єктів, що можуть становити для користувача пошуковий інтерес. Бо часто саме за синонімами та схожою семантикою знаходяться необхідні користувачу дані. Одним з рішень є побудова семантичних мереж з різними видами зв’язків, що можуть забезпечувати перехід по зв’язкам між об’єктами для пошуку даних.

Різні користувачі під одними словами можуть мати різне. Тому постає проблема визначення пошуковою системо, що саме потрібно користувачу і становить для нього пошуковий інтерес. Саме тому, зараз часта практика, що система має певні дані про користувачів для опису їх моделей персони. Для створення такої моделі, можуть застосовуватись:

- попередні пошукові запити користувача з прив’язкою які саме результати він обрав;
- історія відвідувань користувачем мережі, для формування його інтернет особистості;
- вказані самим користувачем дані про себе. Характерна для сервісів з реєстрацією, в які інтегрований пошук;
- аналіз схожих пошуків інших користувачів, для побудови пошукових тенденцій.

Доцільним є попередній семантичний аналіз даних, що становлять базу для пошуку. В такому випадку проводиться аналіз тексту для визначення ключових пам’яток, по яких потім проходить маркування об’єктів бази. Такий підхід забезпечує швидший пошук семантичної інформації та більшу відповідність пошукового результату, запиту користувача.

Кінцевим етапом є вивід результатів пошуку. Зазвичай об’єктами результату виступають, необхідні користувачу з боку алгоритму, частини джерел даних. В такому випадку при обрані користувачем певного об’єкту результату, можуть додаватись посилання на пов’язані дані. У користувача ще на етапі формування

запиту присутня можливість обрати конфігурації формату виводу результату. Також інтерфейс результату може містити опції для форматування результату. Наприклад відсортувати всі знайдені об’єкти по певній полю ознаки, перейти на іншу сторінку результату.

Існують два показники ефективності пошукової системи: пошукова відповідність та пошукова якість. Пошукова відповідність визначається через частку відібраних об’єктів результату, з числа проаналізованих об’єктів. Пошукова якість це частка об’єктів, що підходять під запит користувача з числа відібраних системою для результату. Для збільшення пошукової відповідності використовуються вище описані пошук по синонімам та семантичні мережі. Пошукова якість може бути збільшуватись, якщо використовувати модель персони, аналіз пошукових трендів. Пошукова відповідність так і пошукова якість може бути збільшена при використанні алгоритмів аналізу задоволеності користувачем результатом пошуку. До таких може відноситись як присутність простої опції оцінки пошуку користувачем, так і наприклад аналіз скільки об’єктів користувач переглянув до того моменту як задовольнив свій пошуковий інтерес і відповідно тримав необхідні дані.

Для даного дипломного проекту будемо знаходити 6 видів сутностей. Це персони, локації, дати, назви, числа та позиції. Персонами виступають імена людей у різних форматах. Під локаціями ми розуміємо географічні об’єкти з словами які вказують їх тип (озеро, місто, штат) якщо такі зустрічаються. Назвами є найменування установ, організацій та сутності типу локації в значення об’єктів дії (наприклад Європа прийняла рішення). Позиції це туп сутностей, що описують позицію, що можуть займати сутності типу персони. Тобто їх посади та приналежність. Типом числа виступають сутності, що складаються з цифр. Для класифікації аналізуються додаткові слова, що стоять поряд в тексті з сутностями типу числа.

2.3 Обгрунтування вибору засобів для зберігання семантичних мереж та виконання семантичних запитів (Neo4j, Chither)

В процесі семантичного аналізу даних відбувається побудова семантичних мереж. Вони є системою об’єктів в масивах даних, визначених системою та зв’язків між ними. Наприклад: ці терміни зустрічаються в одному документі, ця сутність про подію прив’язана до цієї сутності часу.

Одним з засобів побудови і зберігання семантичних мереж є система для створення графових баз даних Neo4j. В ній застосвується мова запитів Cypher. Як і в звичаному графі основу складають вершини, що є відображенням об’єктів і ребра, що відображають зв’язки між цими об’єктами (рис. 2.7).

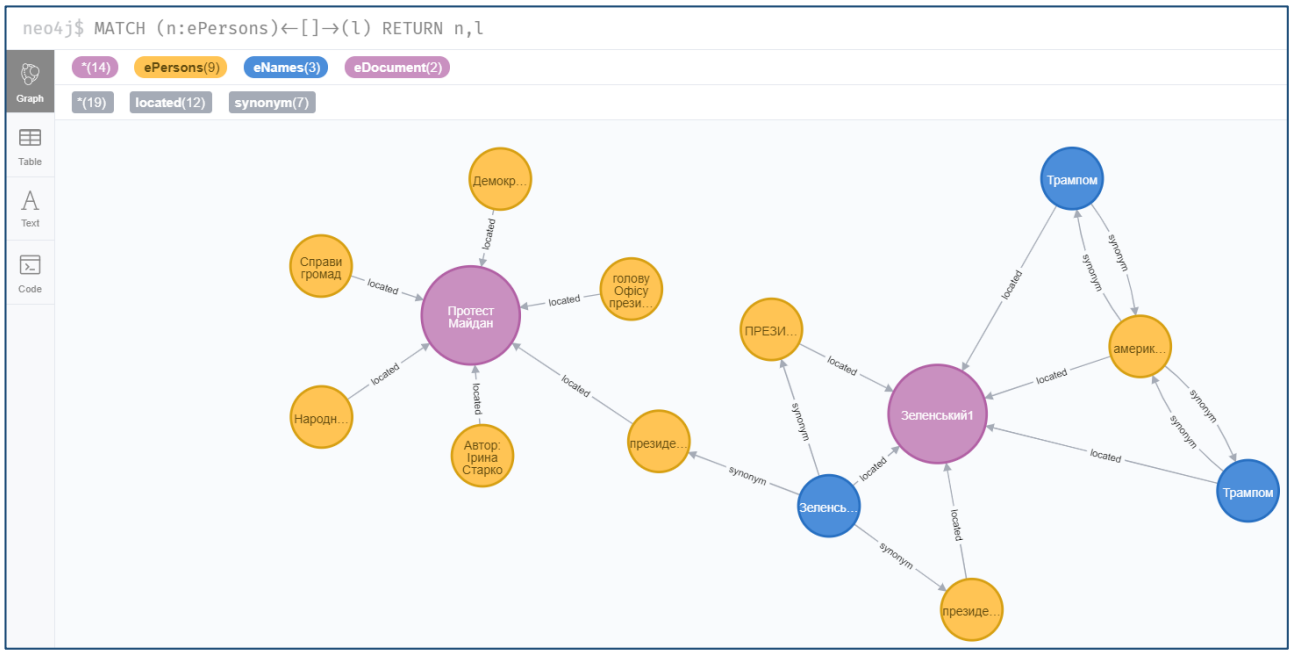


Рис. 2.7 Результат відображення запиту в Neo4j

Графічне відображення даних є більш зручним для людини. Користувач з легкістю може знайти необхідний об’єкт і по вузлам прослідкувати побачити пов’язанні об’єкти.

Проведемо порівняння графової та реляційної бази даних у таблиці 2.1. [15]

Таблиця 2.1 Порівняння Neo4j та реляційної бази даних

Характеристика	Реляційна база даних	Neo4j
Зберігання даних	Зберігання у фіксованих, заздалегідь заданих таблицях із рядками та стовпцями куди підключають данні, часто роз'єднані між таблицями, понижує ефективність запитів.	Структура зберігання графіків із суміжністю без індексів призводить до більш швидких транзакцій та обробки даних для зв’язків даних.
Мова запиту	SQL : Мова запитів, що збільшує складность з кількістю JOIN, необхідних для підключених запитів даних.	Cypher : рідна мова запиту графів, яка забезпечує найбільш ефективний та зручний спосіб опису запитів взаємовідносин.
Виконання запитів	Продуктивність обробки даних страждає від кількості та глибини JOIN (або запитів відносин).	Обробка графіків забезпечує нульову затримку та продуктивність у режимі реального часу, незалежно від кількості чи глибини зв’язків.
Моделювання даних	Модель бази даних повинна бути розроблена за допомогою моделюваннь і переведена з логічної моделі на фізичну. Оскільки типи та джерела даних повинні бути відомі достроково, будь-які зміни потребують тижнів простою для впровадження.	Гнучка, "зручна дошка" модель даних, без невідповідності між логічною та фізичною моделлю. Типи даних та джерела можна додавати або змінювати в будь-який час, що призводить до різко коротших термінів розробки та справжньої гнучкої ітерації.
Підтримка транзакцій	Підтримка транзакцій ACID, необхідна корпоративним програмам для отримання послідовних та достовірних даних.	Зберігає транзакції ACID за цілком цілісні і надійні дані цілодобово - ідеально підходить для постійних глобальних корпоративних програм.

Таблиця 2.1 Порівняння Neo4j та реляційної бази даних (закінчення)

Характеристика	Реляційна база даних	Neo4j
Обробка на масштабі	Масштабування через реплікацію та масштабування архітектури можливо, але дороге. Складні зв’язки даних не збираються в масштабі.	Графічна модель властива масштабам для запитів на основі шаблону. Масштабування архітектури підтримує цілісність даних за допомогою реплікації. Масивні масштаби розширення можливостей із системами IBM POWER8 та CAPI Flash.
Ефективність центру обробки даних	Консолідація сервера можлива, але дорога для масштабування архітектури. Масштабування архітектури дороге з точки зору придбання, використання енергії та часу управління.	Дані та взаємозв'язки зберігаються в сукупності разом із покращенням продуктивності в міру зростання складності та масштабу. Це призводить до консолідації сервера та неймовірно ефективного використання обладнання.

В Neo4j максимальна кількість зв’язків становить 2 в степені 34 мільярди, та 32767 типа зв’язків. Це забезпечує можливість зберігання великих графових моделей звязків. Інтеграція Neo4j дозволяє легко завантажували великі масиви даних з інших БД і швидко перетворювати їх у графи. Вершини поєднуються відносинами, що завжди мають спрямованість. В мові Cypher присутня система автоматичної візуалізації даних, що полегшує читання результату запиту користувачем (рис. 2.8).

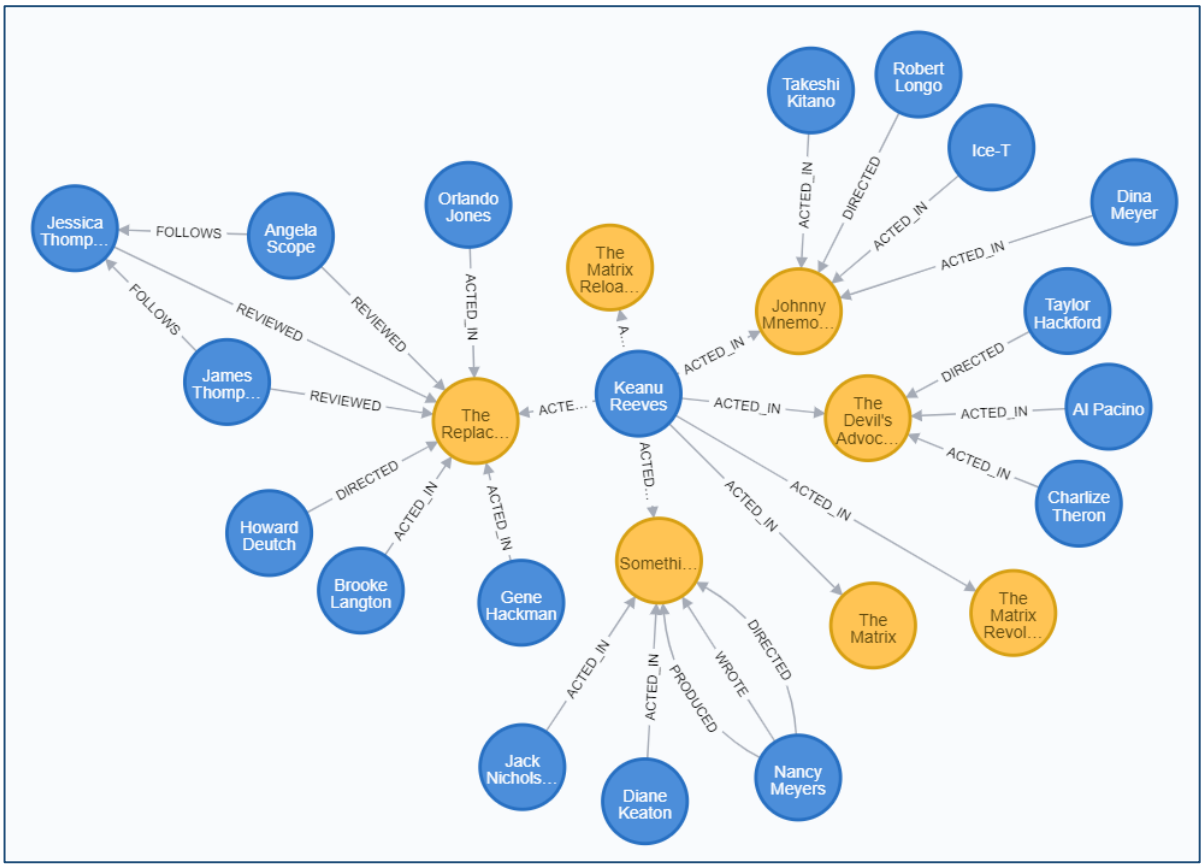


Рисунок 2.8 Візуалізація данних в Neo4j

Дана мова запитів, дозволяє описувати об’єкти та зв’язки на різних рівнях, створюючи система класів та підкласів характеристик. Це дозволяє доповнити вершини та ребра додатковими полями (графічне представлення цього на рис. 2.9) та забезпечує подальші можливості для пошуку. Cypher описує графи, використовуючи специфікацію за зразком - використовується проста форма ASCII-графіки, користувач малює частина графа, його цікавить, за допомогою ASCII символів; вершини беруться в дужки, їх мітки прописуються після «:»; для створення декількох вузлів їх слід перерахувати через »,«; зв'язку відображаються стрілками (-> і <-), а назви зв'язків вказуються всередині квадратних дужок після «:»; властивості вузлів і зв'язків (пари ключ-значення) прописуються в фігурних дужках.

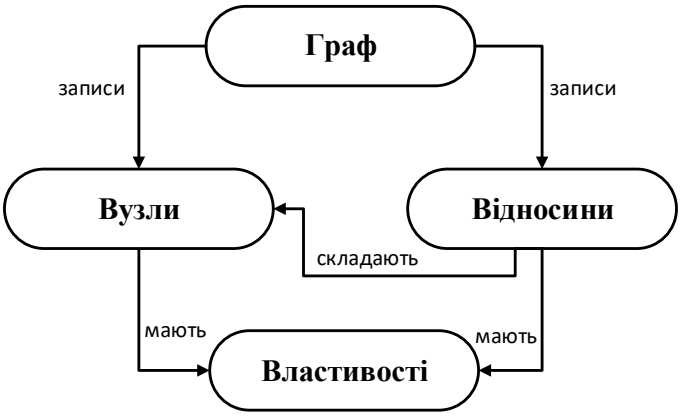


Рис. 2.9 Приклад схеми зв’язків

Інтерфейс програмування додатків для СУБД реалізований для багатьох мов програмування, включаючи Java, Python, Clojure, Ruby, PHP, також реалізовано API в стилі REST.

ВИСНОВКИ ДО РОЗДІЛУ 2

Методи опрацювання запиту полягають в перетворені запита користувача на людській мові в мову зрозумілу машинним алгоритмам. Завантаженні користувачем документи мають проходити попередні перевірку для забезпечення зниження помилок сервісу при опрацюванні тексту. В силу специфіка обробки семантичного запиту кращим рішенням є виділення окремих бази даних під запис документів з опрацьованими файлами та семантичної мережі сутностей архіву документів.

Для запису сутностей ми будем використовувати графічну базу даних Neo4j. Вона має високий рівень візуалізації даних, представляє дані у різних видах та забезпечує швидке перетворення в інші види зберігання баз даних. Дозволяє перетворювати інші формати баз даних у графові моделі. Мова запитів Cypher даної бази даних забезпечує гнучкість запитів для обробки об’єктів бази даних.

РОЗДІЛ 3 РОЗРОБКА ТА ТЕСТУВАННЯ СЕРВІСУ

3.1 Опис роботи сервісу та запиту опрацювання документу

Потрібно розробити сервіс який буде опрацьовувати документи та знаходити в них іменовані сутності для подальшого опрацювання: семантичного пошуку, виводу семантичних структур даних, виявлення додаткової неявної інформації. Була обрана реалізація при якій створювався файл сутностей, що буде прив'язаний до завантаженого документу. Це дозволило більш швидкого опрацьовувати ключові дані документа, працюючі не з текстом документу, а з файлом виявлених іменованих сутностей. На основі цих файлів будувалась семантична мережа, що є базою даних всіх сутностей в архіві документів. Опис моделі сервісу зображений на рис. 3.1.

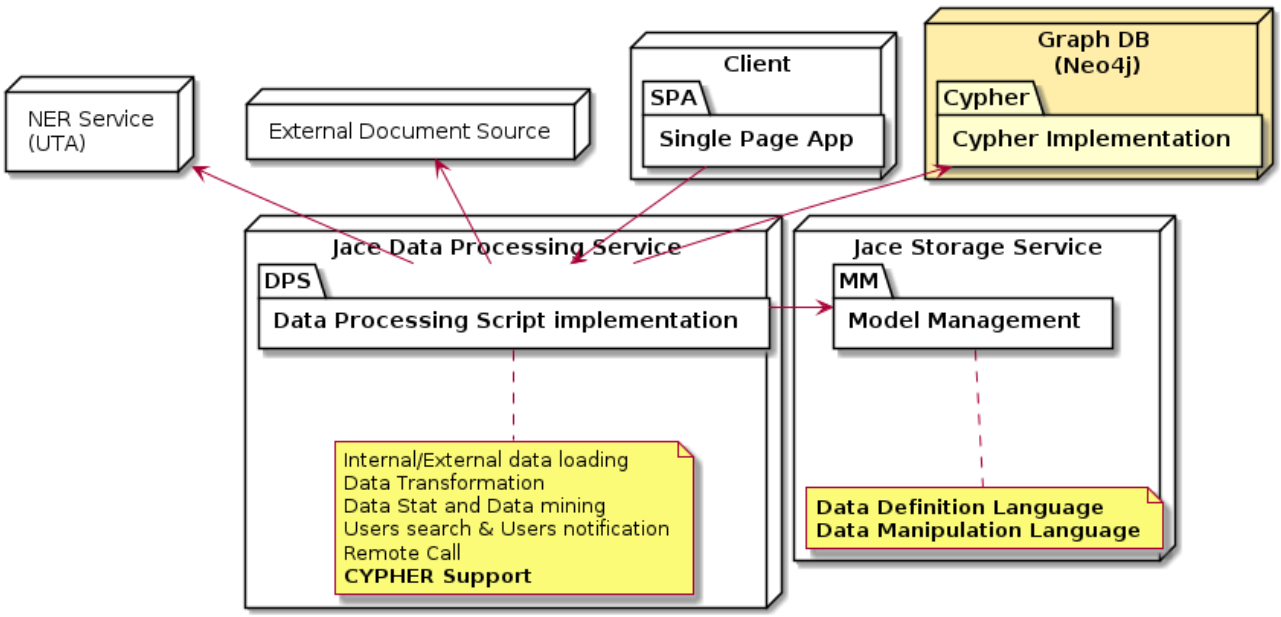


Рисунок 3.1 Модель сервісу

Взаємодія користувача з сервісом відбувається через односторінковий застосунок. Спочатку користувач обирає документ який треба опрацювати, вказує конфігурації та запускає запит на опрацювання. Запит передається в службу обробки даних (Jace [16]), де виконується керування подальшими процесами: обробки даних, перетворення, запису, відправки та отримання повідомлень від

користувачів, формування та відправки результату користувачам. Служба обробки даних записує документ в базу даних та відправляє його текст сервісу з пошуку іменованих сутностей. Сервіс пошуку сутностей, на основі словника сутностей та мови, визначених шаблонів, функцій з обробки виявляє іменовані сутності в тексті, визначає їх поля та властивості і записує в json файл. Також сервіс визначає деякі види зв'язків між сутностями та додає їх в json. Даний файл зберігається за допомогою служби зберігання jace, та прив'язується сервісом до документа в базі даних. Файл становить основу для майбутньої роботи сервіса із сутностями документу та функціями користувача з опрацювання даних. Далі json файл опрацьовується сервісом для доповнення графічної бази даних, де міститься семантична мережа з усіма сутностями та їх зв'язками, у вигляді графа. Для зберігання використовується графова база даних Neo4j. Схема виконання запиту сервісом зображена на рис. 3.2.

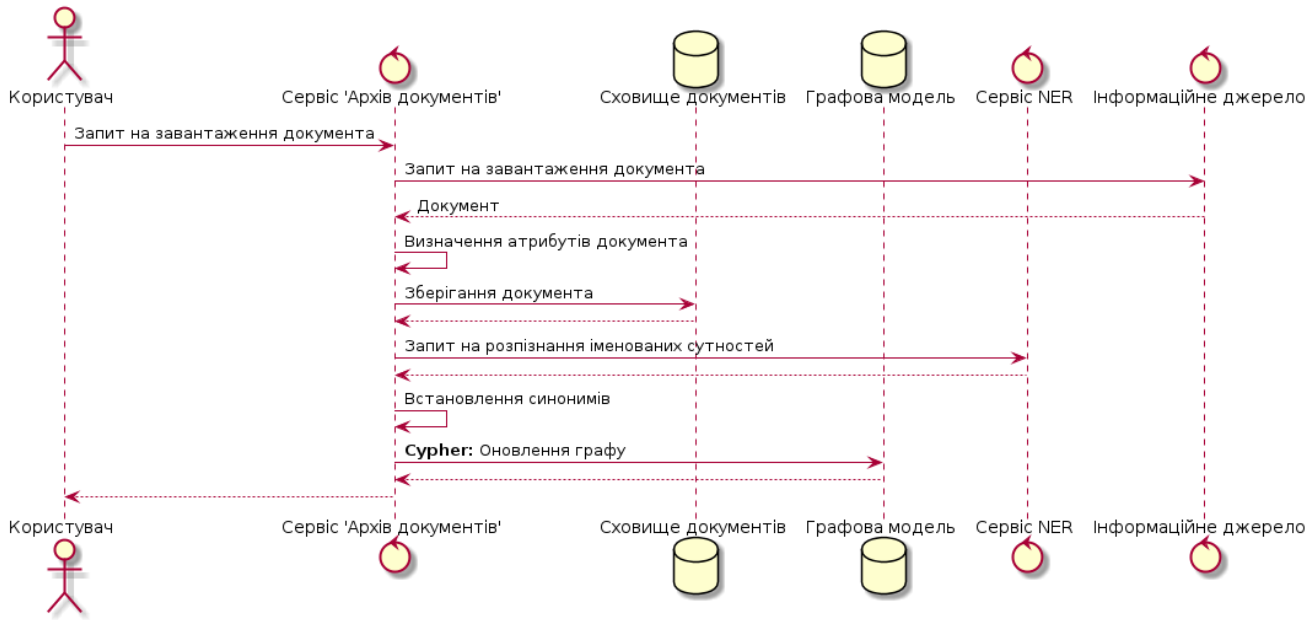


Рисунок 3.2 Схема виконання сервісом запиту користувача

Як видно зі схеми, користувач спочатку завантажує документ для опрацювання. Для цього він відправляє сервісу запит на завантаження документу. Сервіс відправляє запит інформаційному джерелу і у відповідь отримує документ. Після цього користувач може обрати конфігурації обробки документу, заповнити анкету документу. Потім користувач запускає виконання запиту і сервіс починає

обробку для визначення атрибутів документу. Після документ записується в базу даних і відправляється запит сервісу NER для того, щоб він виконав процес пошуку іменованих сутностей в тексті документу. Після опрацювання документу сервіс NER відправляє повідомлення про закінчення роботи разом з json файлом сутностей та зв'язків. Файл результату оброблює сервіс для обновлення графічної бази даних сутностей – семантичної мережі.

3.2 Типи сутностей та зв'язків.

Структура типів сутностей та зв'язків для розпізнавання сервісом та семантичної мережі однакова.

name – обов'язкове поле назви об'єкта. Дане поле присутнє у всіх об'єктах семантичної мережі. Тип string. В семантичні мережі значення цього поля відображається користувачеві.

eDocument – тип об'єктів, що описують опрацьовані документи, що знаходиться в базі архіву документів. В ролі документа може виступати інтернет-сторінка.

Поля властивостей:

link – обов'язкове поле посилання на документ. Для веб сторінок це її URL адреса. Тип string.

category – обов'язкове поле категорії документу. Наприклад: новини, наказ, звіт. Може бути пустим, якщо категорія невизначена. Тип string.

author – обов'язкове поле, де вказується автор, творець документа. Тип string.

pubDate – обов'язкове поле дати створення документа або публікація сторінки. Тип string.

ePersons – тип об’єктів, що є персонами, тобто сутностями, що мають ім’я.

Поля властивостей:

fist – опціональне поле, що містить ім’я персони. Тип string.

last – опціональне поле прізвища персони. Тип string.

middle – опціональне поле імені по батькові персони. Тип string.

eDates – тип об’єктів, що є датами (роки, місяці, числа місяців).

Поля властивостей:

day – опціональне поле числа місяця. Тип int.

month – опціональне поле назви місяця. Тип string.

year – опціональне поле року. Тип int.

current_era – обов’язкове поле, що вказує якої ери дана дата. “True” наша ера, “False” – до нашої ери. Тип string. За замовчуванням дата вважається нашою ерою, якщо в її складі немає специфічних слів типу: «до нашої ери», «до н.е.», «до Різдва Христового», «до Р.Х.», «до року Божого», «до р. Б.», «BC», «BCE». Причому регістр слів в таких шаблонах немає значення. Це було зроблено для того, щоб якщо в тексті буде допущена помилка в регістрі, сервіс зміг розпізнати, що дана дата підходить під шаблон дат до нашої ери.

eLocations – тип об’єктів, що є локаціями (географічні об’єкти). Розпізнаються за рахунок словника географічних об’єктів та шаблонів граматики прописаних в програмних модулях.

ePositions – об’єкти, що є назвами позицій персон, тобто їх посадами, приналежністю (президент, депутат, солдат, громадянин).

eNames – тип об’єктів, що є назвами установ або сутностей в значенні об’єкта виконання дії.

eNumeric – тип об’єктів, що є числами або виражають числову ознаку (число, відсоток, кількість, порядок).

category - обов’язкове поле, що вказує якого виду є число (відсоток, кількість, номер, вік об’єкта).

Між об’єктами існують зв’язки. Існує 5 видів зв’язків: *located, position, synonym, homonym, author*.

located – тип зв’язку, що встановлюється від об’єкта сутності в напрямі об’єкта документу. Даний зв’язок показує, що сутність знаходиться в документі. Так на семантичній мережі видно, які сутності зустрічаються в документі та в яких документах знаходиться певна сутність.

position – тип зв’язку, що встановлюється від сутності персони (тип об’єкта *ePersons*) в напрямі сутності посади, приналежності (тип об’єкта *ePosition*). Має поле *document* типу string, що позначає в контексті якого документа дана персона займає дану посаду, позицію.

synonym – тип зв’язку, що встановлюється між об’єктами, що є одним і тим же. Даний зв’язок будується в двох напрямах між об’єктами.

homonym – тип зв’язку, що встановлюється у двох напрямах між об’єктами, що мають однакову назву, але відносяться до різних типів. Тобто мають різне семантичне значення.

author – тип зв’язку між автором, творцем документу та документом.

Для заповнення полів об’єкта типу eDocument використовується попередня анкета конфігурацій, в якій користувач може вказати значення цих полів. Якщо

користувач її не заповнив, або деякі поля не були заповненні, сервіс має декілька функцій для автоматичного заповнення полів. Для суто файлів документів поле назви об’єкта eDocument буде отримано з назви завантаженого файлу. Поле автора документа початково приймає значення імені користувача, що його завантажив. Поле дати публікації, є датою та часом завантаження документа в систему. Поле категорії з початку є невизначеною, якщо його не вказав користувач.

У випадку якщо документом виступає інтернет сторінка і користувач вказав посилання на неї, може опрацьовуватись наявна RSS-стрічка (рис. 3.3). RSS – це спеціальний формат, що описує стрічки новин, статей та оновлень блогу. В ньому

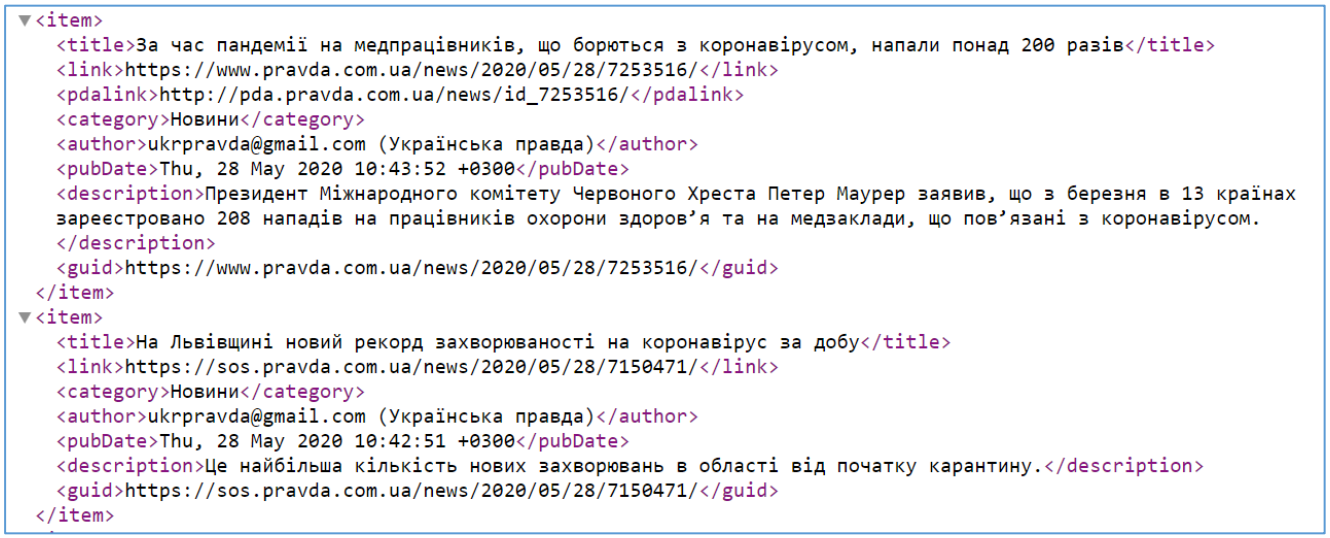


Рисунок 3.3 Приклад частини RSS

описана структура списку сторінок по ключовим полям. Зазвичай посилання на ресурс, дата створення, назва сторінки, адреса в архіві серверу, ключові теги та інше.

Як видно з рис. 3.3 тут містяться поля (title, link, category, author, pubDate), що співпадають з полями опису об’єкту типу eDocument. Значення полів витягуються з RSS та перезаписують значення полів об'єкту документа. Для різних сайті може відрізнятись специфіка RSS. Тому існує декілька шаблонів розпізнавання синонімічних тегів полів. Поле *author* може заповнюватись по вмісту текста документа. Якщо модуль розпізнавання в початковому або кінцевому сегменті

знаходить сутності ePosition (зі значенням «Автор», «Автори» «Авторський колектив», «Укладач», «Розробник») та ePersons, що знаходяться разом по одному із шаблонів:

[ePosition: «Автор»] [ePersons]

[ePosition: «Автор»] «:» [ePersons]

[ePosition: «Укладач»] [ePersons]

[ePosition: «Укладач»] «:» [ePersons]

[ePosition: «Розробник»] [ePersons]

[ePosition: «Розробник»] «:» [ePersons]

[ePosition: «Автори»] [ePersons1] [ePersons2]… [ePersonsN]

[ePosition: «Автори»] «:» [ePersons1] [ePersons2]… [ePersonsN]

[ePosition: «Авторський колектив»] [ePersons1] [ePersons2]… [ePersonsN]

[ePosition: «Авторський колектив»] «:» [ePersons1] [ePersons2]… [ePersonsN]

Так як у нас подокументне завантаження у базу даних документів та графову базу даних сутностей, зв'язки типи *located* встановлюються між усіма сутностями, що знаходяться в документі та самим документом. Спочатку створюється об'єкт документа з полями властивостей, значення який визначаються по описаних вище методам. Одразу після створення об'єкту йде створення зв'язку *located* між об'єктом сутності та об'єктом документа де ця сутність знаходиться. При цьому використовується команда пошуку об'єкта сутності та об'єкта документа по їх індефікатору, яка передає знайдені об'єкти в шаблон запиту створення зв'язку.

Тип зв'язку *position* встановлюється на етапі розпізнавання сутностей, якщо в тексті разом знаходяться сутності типу ePerson та ePosition згідно одного з двох шаблонів: [ePosition] [ePersons], [ePersons] [ePosition]. В різних документах певна персона може бути прив'язана до різних позицій. Тому для індефікування в контексті якого документа персона займає певну позицію, зв'язок *position* має поле document де записується ім'я документа.

Тип зв’язку *synonym* встановлюється між об’єктами одного типу, які є один і тим самим, на етапі розпізнавання іменованих сутностей. Цей зв’язок буде встановлюватись між сутностями одного документа. Для цього проходить порівняння збігів полів властивостей сутностей:

- для типу eDates повного збігу полів *day, month, year, current_era.* Наприклад «28 червня» та «28.08» (співпадають поля *day* = 28, *month* = «червень» та *current_era* = «True») *;*
- для типу ePersons збігу декількох полів з *fist, last, middle.* Наприклад «Порошенко» та «Петро Порошенко» (співпадає поле *last*);
- для типів сутностей ePositions та eNames порівнюється ступінь співпадіння поля name. У словнику дані сутності вказуються разом з їх абревіатурами та скороченнями.

Зв’язок встановлюється в двох напрямах. Тобто при виявлені синонімічної сутності зв’язок будується від сутності в напрямі сутності які було розпізнано як синонім, та в зворотньому напрямі. Призначення цього зв’язку в поєднанні сутностей, що виступають одним об’єктом. Це дозволяє опираючусь на ці зв’язки знаходити всі назви одного об’єкта. Наприклад коли потрібно знайти всі документи, де згадується певна персона, в запиті вказується будь-яке з його назв. Знаходяться всі об’єкти семантичної мережі з якими даний об’єкт має зв’язок типу *synonym.* Потім знаходяться всі об’єкти типу eDocument з якими мають зв’язки об’єкти синоніми.

Зв’язок *homonym* будується вже в самі й семантичній мережі. Бо сутності, що мають однакове поле *name* але різні типи об’єкта сутності, можуть знаходить в різних документах. Для реалізації цього використовується Cypher запит, що шукає всі об’єкти з однаковими назвами але різними типи, якщо між ними немає зв’язку *homonym*. Результати пошуку передаються для генерування запитів створення зв’язків.

Зв’язок *author* будується після додавання сутностей виявлених в документі до семантичної мережі. Для цього береться значення поля *author* об’єкта даного документа. По цьому значенню проводиться пошук об’єктів сутностей, що мають таке ж значення поля *name*. Якщо об’єкт сутності знайдений, він передається запиту для створення зв’язку з об’єктом документу.

3.3 Генерування семантичної мережі.

Для зберігання виявлених сутностей використовували графову база даних Neo4j. Сутності зберігались у вигляду вершин графів, зв’язки між сутностями у вугляді ребер. В результаті на виході в базі була записана семантична мережа (рис.3.4), для подальшого опрацювання. Така форма запису даних результату, дозволяє їх візуалізувати для кращого розуміння користувачем.

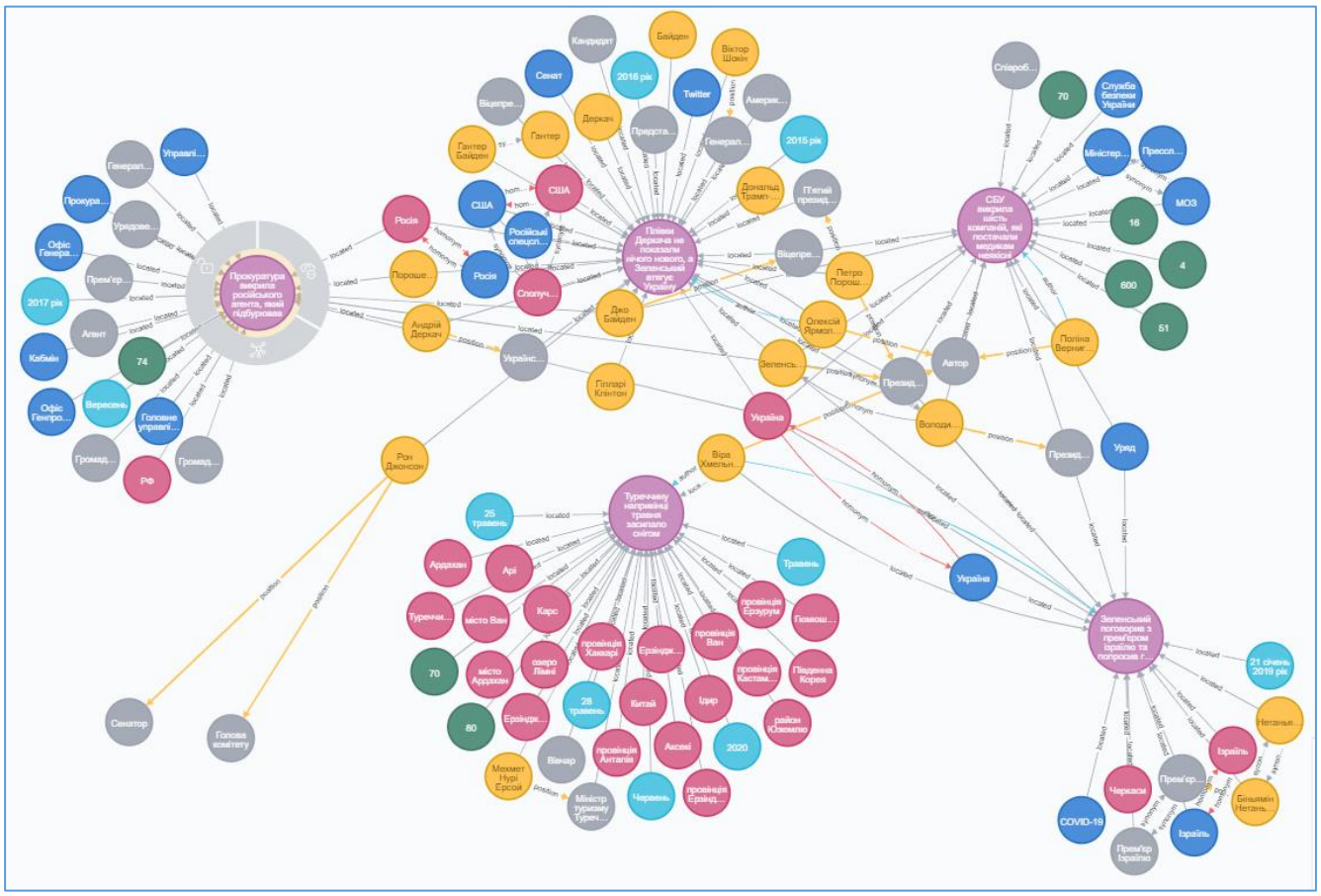


Рисунок 3.4 Вигляд семантичної мережі в Neo4j

Для перетворення об’єктів записанних в файлі «*.json» написали програмний модуль, генерування запитів Cypher. Модуль читає файл і створює списки сутностей та зв’язків між ними, з полями властивостей. Відмітимо, що якщо якась сутність, або зв’язок вже записані в базу Neo4j, наприклад зустрічалась в раніше опрацьованих і доданих в базу документах, то вони не будуть додані в список. Елементи списку сутностей передаються в скрипт з шаблоном Cypher створення об’єкта. І даний скрипт передається на виконання. Так послідовно проходячи список, база даних наповнюється новими об’єктами-сутностей. Після запису всіх сутностей списку в базу, починається етап створення зв’язків між ними. Для цього елементи списку зв’язків передаються в скрипт шаблону Cypher створення зв’язку між об’єктами. Відмітимо, що в списку зв’язків зберігаються всі зв’язки крім *located,* бо їх створення йде одразу після створення об’єкту. В результаті семантична мережа стає доповненою сутностями та зв’язками, що містяться в опрацьованому документі.

В Neo4j присутня функція імпорту даних з json, cvs та інших баз даних за допомогою запитів. Проте в залежності від файлу запит може містити специфічні поля. Наперед встановити яким має бути запит складно. Тому був обраний метод генрування запитів створення кожного об’єкту та зв’язку.

Після додавання документу в семантичну мережу, користувач може взаємодіяти з мережою за допомогою запитів Cypher.

3.4 Ін’єкції Cypher через dps

Взаємодію з базу реалізували за допомогою ін’єкцій Cypher. При цьому ми використовуємо наперед заданий шаблон ін’єкції Cypher, та синтаксис для опису запиту. Потрібно знайти найкоротші шляхи між ePersons {name:"Зеленський"}) та :ePersons {name:"Порошенко"} через всі вузли об’єктів та зв’язки між ними. Для отримання даних із бази даних мережі сутностей використовуємо команду service.cypher з параметами за замовчуванням. Даний уривок коду показаний на рисунку 3.5

```
<?cypher
MATCH p=allShortestPaths(
        (:ePersons {name:"Зеленський"})-[*]-(:ePersons {name:"Порошенко"})
    )
    RETURN distinct p
?>

service.cypher()
```

Рисунок 3.5 Уривок коду з ін'єкцією Cypher для пошуку найкоротшого шляху між об'єктами

Врезультаті виконання отримується json (рис. 3.6), що містить всі об'єкти та зв'язки включаючі їх поля. Там буде міститись вся структура, що утворює граф найкоротших шляхів між вказаними об'єктами. Цей файл результату може використовуватись іншими модулю для опрацювання та виводу у необхідній формі За допомогою вбудованої інтеграції Neo4j дозволяє перетворювати цю структуру в інші формати і візуалізувати.


```
        "identity": 83,
        "labels": [
          "ePersons"
        ],
        "properties": {
          "last": "Порошенко",
          "name": "Порошенко"
        }
      },
      "segments": [
        {
          "start": {
            "identity": 1,
            "labels": [
              "ePersons"
            ],
            "properties": {
              "last": "Зеленський",
              "name": "Зеленський"
            }
          },
          "relationship": {
            "identity": 120,
            "start": 1,
            "end": 85,
            "type": "located",
            "properties": {}
          },
          "end": {
            "identity": 85,
            "labels": [
              "eDocument"
            ],
            "properties": {
              "name": "Плівки Деркача не показали нічого нового, а Зеленський втягує Україну у конфлікт із США – WP",
              "link": "https://tsn.ua/politika/
              plivki-derkacha-ne-pokazali-nichogo-novogo-a-zelenskiy-vtyaguye-ukrayinu-u-konflikt-iz-ssha-wp-1553886.html",
              "category": "Новини",
              "pubDate": "Tue, 26 May 2020 12:54:00 +0300",
              "author": "Олексій Ярмоленко"
            }
          }
```


Рисунок 3.7 Вигляд для серверу результату виконання запиту ін'єкцією Cypher.

Як буде виглядати результат для користувача показано на рисунку 3.8. Користувач отримує граф, що відображає результат та моє можливість переглядати властивості кожного об’єкту та зв’язку наводячи на них.

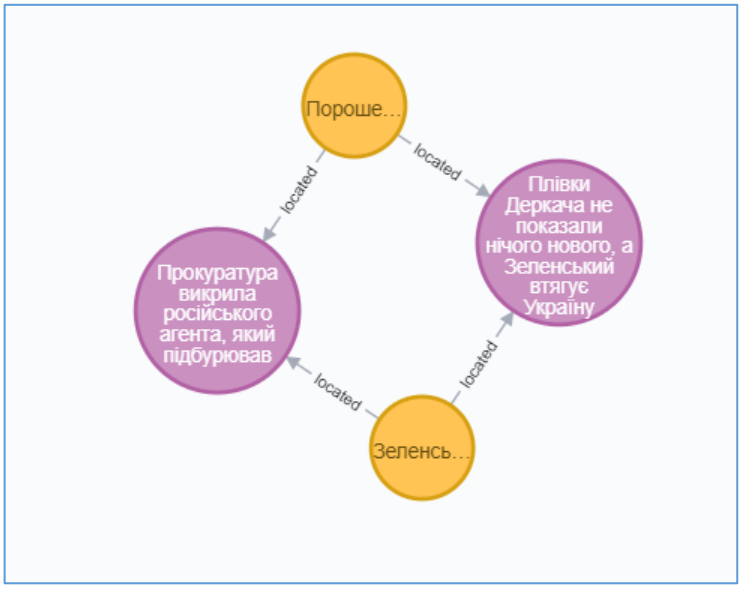


Рисунок 3.8 Вигляд для користувача результату виконання запиту ін’єкцією Cypher.

Нам потрібно забезпечити динамічність шаблону для запиту. Для цього пропишемо зміні, які прийматимуть значення імен об’єктів для пошуку, введених користувачем. Використаємо Javascript-вставку де зміні будуть приймати значення. В шаблоні у Cypher-вставці будуть вписані вже не імена об’єктів, а зміні де зберігається ці імена. За рахунок такого коду (рис. 3.9) забезпечена можливість динамічної генерації запитів Cypher під різні користувацькі запити даного типу.

```
    <?javascript
    $scope.params = {"p1":"Зеленський","p2":"Порошенко"}
 ?>

<?cypher
    MATCH p=allShortestPaths(
        (:Person {name:"${p1}"})-[*]-(:Person {name:"${p2}"})
    )
    RETURN distinct p
?>

set("query")

service.cypher(
    query:<?
        _.template($scope.query)({p1:$scope.params.p1, p2:$scope.params.p2})
    ?>
)
```

Рисунок 3.9 Уривок коду для підстановки об’єктів запитів Cypher.

Використовуємо команду service.cypher із query, значення якого є js-injection. Застосовується функція lodash для отримання цього значення. В результаті отримується json для сервісу та візуалізоний граф для користувача. Вони мають таку ж структура як і в прикладах вище, але відповідають об’єктам які є значеннями змінних. Для забезпечення більшої динамічності запиту, типу об’єктів також можна передавати через зміні. При цьому з’являється можливість отримувати граф зв’язків між будь-якими двома вершинами вершинами. У даному випадку граф найкоротших шляхів.

Сервіс Neo4j при роботі у вебі, відображає результат у вигляді посилань на об’єкти без розгортування їх структуру. Це викликано тим, що при розгортуванні можуть з’являтись рекурсивні посилання і сервісу буде складно зрозуміти, як правильно відобразити результат. Для того, щоб відображались самі об’єкти використовували populate в функції service.cypher (рис 3.10). Значеннями populate є об’єкт або масив об’єктів типу string. Кожен об’єкт визначає схему шляху властивостей об’єкту результату. Так *data[*][*].[relationships, nodes].** визначає всі властивості всіх об’єктів сутностей та зв’язків між ними, що знаходять в структурі результату запиту.

```
<?cypher
MATCH p=allShortestPaths(
        (:ePersons {name:"Зеленський"})-[*]-(:ePersons {name:"Порошенко"})
    )
    RETURN distinct p
?>

service.cypher(
    populate:[
            "data[*][*].[relationships, nodes].*",
            "data[*][*].relationships.[start,end].*"
    ]
)
```

Рисунок 3.10 Уривок коду для розгортування структури об'єктів по їх посиланням в результатах запитів Cypher.

В результаті ми отримуємо розгорнуту структуру об'єктів, що дозволяє відобразити результат у більш наглядному та зрозумілому вигляді для користувача. На рисунку 3.11 приведено порівняння json файлів результатів без розгортування структуру та зі розгортуванням.


```
{
    "columns": [
        "p"
    ],
    "data": [
        [
            {
                "relationships": [
                    "https://hobby-nlhgecabchbdgbkegbhaepel.dbs.graphenedb.com:24780/db/data/relationship/144",
                    "https://hobby-nlhgecabchbdgbkegbhaepel.dbs.graphenedb.com:24780/db/data/relationship/145",
                    "https://hobby-nlhgecabchbdgbkegbhaepel.dbs.graphenedb.com:24780/db/data/relationship/17",
                    "https://hobby-nlhgecabchbdgbkegbhaepel.dbs.graphenedb.com:24780/db/data/relationship/21"
```

```
{
    "columns": [
        "p"
    ],
    "data": [
        [
            {
                "relationships": [
                    {
                        "extensions": {},
                        "metadata": {
                            "id": 144,
                            "type": "ACTED_IN"
                        },
                        "property": "https://hobby-nlhgecabchbdgbkegbhaepel.dbs.graphenedb.com:24780/db/data/relationship/144/properties/{key}",
                        "start": "https://hobby-nlhgecabchbdgbkegbhaepel.dbs.graphenedb.com:24780/db/data/node/126",
                        "self": "https://hobby-nlhgecabchbdgbkegbhaepel.dbs.graphenedb.com:24780/db/data/relationship/144",
                        "end": "https://hobby-nlhgecabchbdgbkegbhaepel.dbs.graphenedb.com:24780/db/data/node/122",
                        "type": "ACTED_IN",
                        "properties": "https://hobby-nlhgecabchbdgbkegbhaepel.dbs.graphenedb.com:24780/db/data/relationship/144/properties",
```


Рисунок 3.11 Вигляд файлу json результату а) стандартний, без розгортування об'єктів б) із розгортуванням структуру об'єктів результату.

На рис. 3.11 а виділено фрагмент, що на рис. 3.11 б має розгорнуту структуру з основними полями та властивостями. Значеннями деяких є посилання по яким сервіс витягує об'єкти для відображення користувачеві.

3.5 Запити для роботи з семантичною мережею.

Запити роботи з семантичною мережею базуються на двох основних командах Cypher:

MATCH *(<шаблон>)* – для пошуку по шаблону. Можуть бути передані декілька незалежних шаблонів для пошуку.

RETURN *<зміна об'єктів пошуку>* – для виводу результатів.

Першим запитом для роботи із графовою базою даних є виведення всієї семантичної мережі. Для цього застосуємо команду пошуку всіє об'єктів та їх виводу в результат – ***MATCH*** *(p)* ***RETURN*** *p*. В результаті ми отримаємо граф (рис. 3.12) всіх сутностей та зв'язків між ними, що записані в базу даних.

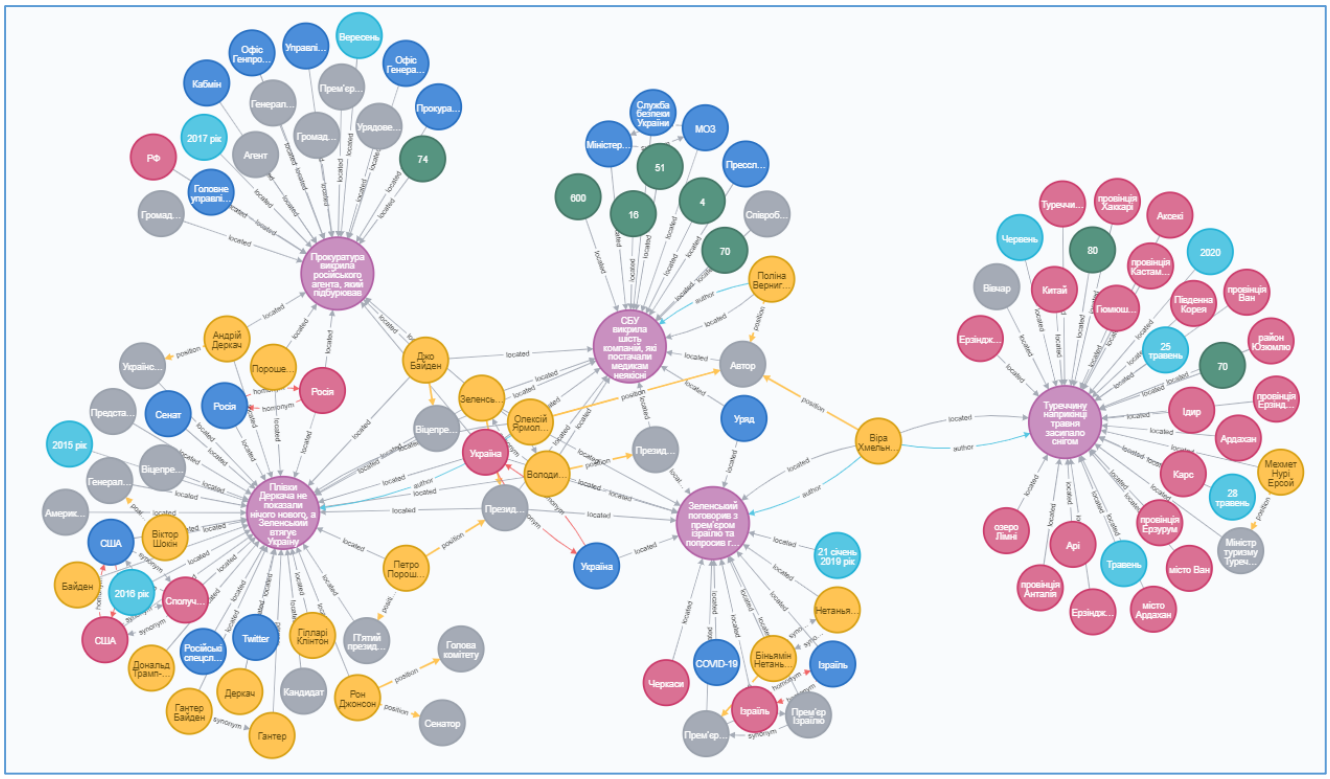


Рисунок 3.12 Граф семантичної мережі

Потрібно вивести всі сутності, що зустрічаються в документі. Для цього в команді MATCH вказуємо шаблон пошуку по об'єктам зі зв'язком і в одному з об'єктів вказуємо документа у відповідному полі. Ми можемо знайти документ по його назві, id, інших полях або їх об'єднанню. Результат пошуку передамо команді

виводу. На виводі отримуємо граф всіх сутностей та зв'язків (рис. 3.13), що зустрічаються в документі. Команди можна об'єднувати в один запит. Для даного прикладу застосували команду:

MATCH (a:eDocument {name : *'Плівки Деркача не показали нічого нового, а Зеленський втягує Україну у конфлікт із США – WP'*})-[]-(b)

RETURN a,b

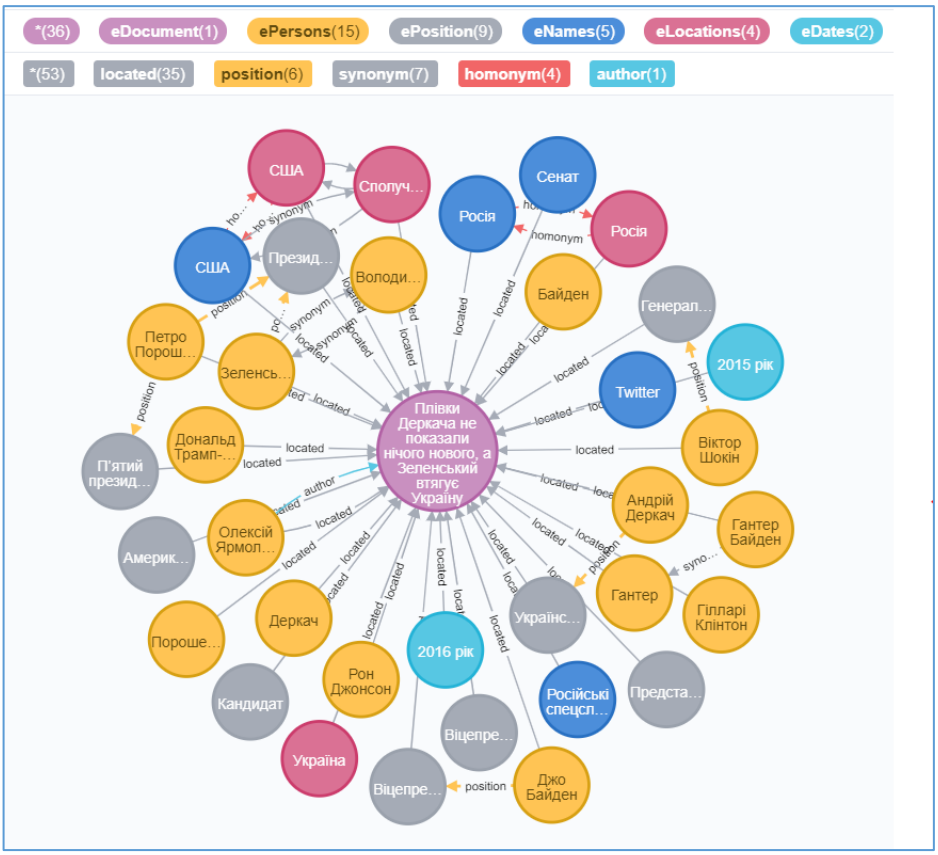


Рисунок 3.13 Граф сутностей, що знаходяться в документі

На панелі зверху отримуємо інформацію про наявні в результаті типи об'єктів та зв'язків з вказанням їх кількості. Натиснувши на об'єкт або його тип з'явиться нижня панель де можна змінювати параметри відображення (колір, розмір, поле яке виступатиме підписом елемента). Це дозволяє забезпечити більш зручний вигляд результату.

Для виводи всіх об'єктів, що повязанні з певною сутністю та її синонімів використаємо шаблони подвійних зв'язків. Наступний запит виводить всі об'єкти,

що пов’язанні з сутністю «Володимир Зеленський» та її синонімами. Приклад запиту:

MATCH (a:ePersons {name:"*Володимир Зеленський*"})-[r:synonym]-(b)-[]-(c)
MATCH (a)-[]-(d) **return** a,b,c,d

Отримали граф (рис. 3.14) документів де згадується персона Зеленського та позиції, які дана персона займає.

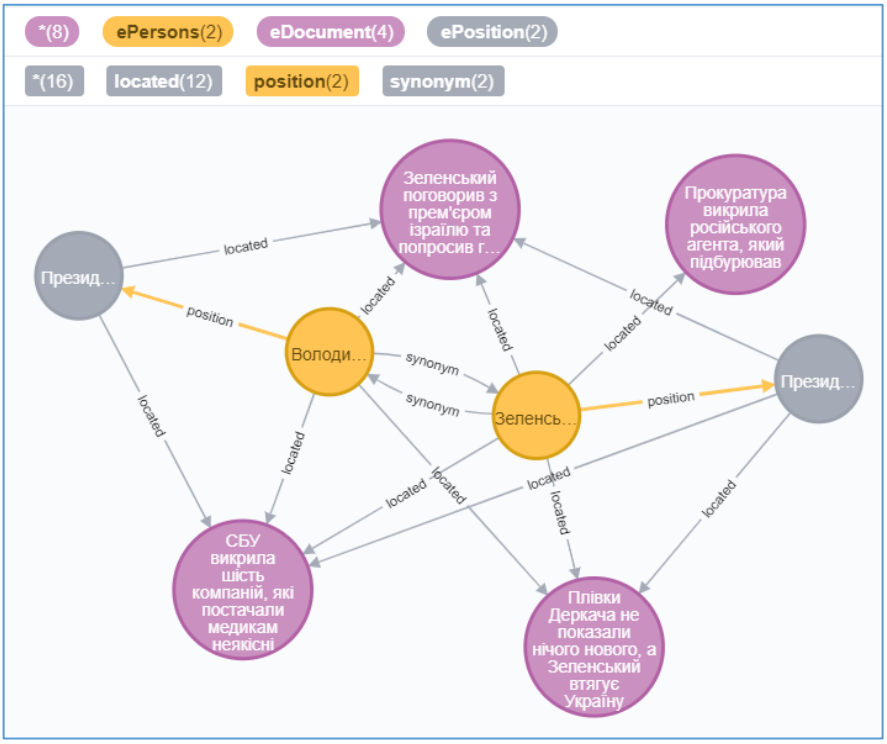


Рисунок 3.14 Відповідь сервісу на запит пошуку пов’язаних сутностей з персоною «Володимир Зеленський»

Для додавання семантичної мережі до інших баз даних існує команда створення файлу запитів створення об’єктів, що входять у результата запиту. Потрібно ввести ввести команду:

CALL apoc.export.cypher.all("Document.cypher", { format: "plain", useOptimizations: {type: "UNWIND_BATCH", unwindBatchSize: 20}})

YIELD file, batches, source, format, nodes, relationships, properties, time, rows, batchSize

RETURN file, batches, source, format, nodes, relationships, properties, time, rows, batchSize;

В результаті створюється файл *Document.cypher* (рис. 3.15), що містить всі команди для створення даної семантичної мережі. Далі потрібно копіювати вміст файлу в поле вводу запитів та запустити виконання.

Document.cypher

```
CREATE CONSTRAINT ON (node:`UNIQUE IMPORT LABEL`) ASSERT (node.`UNIQUE IMPORT ID`) IS UNIQUE;
UNWIND [{_id:3, properties:{name:"Співробітники Служби безпеки України"}}, {_id:5, properties:{name:"Президент"}},
{_id:8, properties:{name:"Президент України"}}, {_id:19, properties:{name:"Автор"}}, {_id:44,
properties:{name:"Вівчар"}}, {_id:49, properties:{name:"Міністр туризму Туреччини"}}, {_id:53, properties:{name:"Прем'єр
Ізраїлю"}}, {_id:55, properties:{name:"Прем’єр-міністр Ізраїлю"}}, {_id:65, properties:{name:"Агент"}}, {_id:66,
properties:{name:"Урядовець Кабміну"}}, {_id:67, properties:{name:"Генеральний прокурор"}}, {_id:68,
properties:{name:"Громадянин Російської Федерації"}}, {_id:74, properties:{name:"Громадянин України"}}, {_id:75,
properties:{name:"Прем’єр-міністр України"}}, {_id:84, properties:{name:"Віцепрезидент США"}}, {_id:87,
properties:{name:"П’ятий президент України"}}, {_id:91, properties:{name:"Американці"}}, {_id:94,
properties:{name:"Кандидат"}}, {_id:99, properties:{name:"Український депутат"}}, {_id:104, properties:{name:"Генеральний
прокурор України"}}] AS row
CREATE (n:`UNIQUE IMPORT LABEL`{`UNIQUE IMPORT ID`: row._id}) SET n += row.properties SET n:ePosition;
UNWIND [{_id:107, properties:{name:"Сенатор"}}, {_id:108, properties:{name:"Голова комітету"}}, {_id:111,
properties:{name:"Віцепрезидент"}}, {_id:112, properties:{name:"Представники адміністрації Трампа"}}] AS row
CREATE (n:`UNIQUE IMPORT LABEL`{`UNIQUE IMPORT ID`: row._id}) SET n += row.properties SET n:ePosition;
UNWIND [{_id:2, properties:{name:"МОЗ"}}, {_id:4, properties:{name:"Пресслужба Офісу президента"}}, {_id:7,
properties:{name:"Служба безпеки України"}}, {_id:10, properties:{name:"Міністерство охорони здоров’я"}}, {_id:11,
properties:{name:"Уряд"}}, {_id:57, properties:{name:"Україна"}}, {_id:58, properties:{name:"Ізраїль"}}, {_id:59,
properties:{name:"COVID-19"}}, {_id:64, properties:{name:"Прокуратура"}}, {_id:69, properties:{name:"Офіс Генерального
прокурора"}}, {_id:71, properties:{name:"Головне управління Генштабу ЗС РФ"}}, {_id:76, properties:{name:"Кабмін"}},
{_id:79, properties:{name:"Управління Служби безпеки України"}}, {_id:80, properties:{name:"Офіс Генпрокурора"}},
{_id:90, properties:{name:"США"}}, {_id:92, properties:{name:"Росія"}}, {_id:98, properties:{name:"Twitter"}}, {_id:100,
properties:{name:"Російські спецслужби"}}, {_id:110, properties:{name:"Сенат"}}, {_id:115,
properties:{name:"Німеччина"}}] AS row
CREATE (n:`UNIQUE IMPORT LABEL`{`UNIQUE IMPORT ID`: row._id}) SET n += row.properties SET n:eNames;
```

Рисунок 3.15 Файл Document.cypher

Також ми можемо копіювати частини семантичної мережі за допомогою зміненою команди. Перед даним запитом потрібно буде додати команду пошуку об’єктів структури яку ви хочете копіювати. Ми можемо провести об’єднання двох баз даних, якщо у створеному файлі команд створення змінимо команди CREATE на MERCE. Після чого запустити отриманий набір команд в консолі введення запитів другої бази даних.

ВИСНОВКИ ДО РОЗДІЛУ 3

У даному розділі проведено розробку сервісу. Приведено опис роботи сервісу, розподілу функцій та етапів виконання модулями сервісу. Описано принципи роботи сервісу з розпізнавання іменованих сутностей та методів за допомогою яких це реалізується.

Розроблено структури типів сутностей, зв’язків та їх полів, що забезпечує виконання подальших завдань з обробки отриманих даних та їх аналізу. Створення об’єктів семантичної мережі та подальша робота з нею базується на використанні запитів мови Cypher напряму і з використанням вставок в коді програми. Візуалізація даних, що лежить в основі графічної бази даних, полегшує читання результатів користувачем та краще розуміння вмісту текстів опрацьованих документів.

ВИСНОВКИ

У даному дипломному проекті проведено розробку сервісу з інтелектуального опрацювання документів за допомогою використання методів розпізнавання іменованих сутностей. Для цього проведено аналіз наявної інформації по задачам NLP, видам пошуку та принципам документообігу для різних типів установ. Обрано семантичний пошук, я вид пошуку, що забезпечує високу відповідність результатів пошуковому запиту та здатний знаходити додаткову інформацію, що може бути корисна для користувача.

На основі наявних рішень і специфікації завдань, які потрібно було вирішити, обрано реалізацію, що забезпечує високий рівень візуалізації завдання і велику кількість можливостей з подальшого опрацювання отриманої семантичної мережі сутностей. Для зберігання семантичної мережі обрано графову базу даних Neo4j, що має можливості з інтеграції інших видів баз даних та синхронізації різних версій однієї бази даних.

Отриманий сервіс можна використовувати як базу для зберігання електронних документів з можливістю інтелектуального опрацювання вмісту документів, як сервіс для проведення семантичного аналізу. Наявність алгоритмів побудови зв’язків між виявленими сутностями дозволяє користуватись сервісом для виявлення семантичних зв’язків у тексті.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Документообіг як складова документного забезпечення управлінської діяльності організацій [Електронний ресурс] – Режим доступу до ресурсу: https://sites.google.com/site/dokumentoobigvustanovi/home/dokumentoobig-ak-skladova-dokumentnogo-zabezpecenna-upravlinskoie-dialnosti-organizacij.

2. Электронный документооборот как способ оптимизации бизнес-процессов [Електронний ресурс] – Режим доступу до ресурсу: https://www.kp.ru/guide/ielektronnyi-dokumentooborot-na-predprijatii.html.

3. Види ЕДО та який варіант документообігу на вашому підприємстві? [Електронний ресурс] – Режим доступу до ресурсу: https://intelserv.net.ua/blog/material/id/158.

4. Електронний архів [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/%D0%95%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%B9_%D0%B0%D1%80%D1%85%D1%96%D0%B2

5. Вірьовкіна Н. М. [Східноєвропейський університет економіки і менеджменту, Україна] «Електронний архів як засіб зберігання та пошуку документальної інформації

6. Поисковые системы Интернета: Яндекс, Google, Rambler, Yahoo. Состав, функции, принцип работы [Електронний ресурс] – Режим доступу до ресурсу: https://www.seonews.ru/masterclasses/poiskovye-sistemy-interneta-yandeks-google-rambler-yahoo-sostav-funktsii/

7. Rambler’s Top100 [Електронний ресурс] – Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/Rambler%E2%80%99s_Top100

8. Основные виды поисковых систем [Електронний ресурс] – Режим доступу до ресурсу: https://v-mire.net/osnovnye-vidy-poiskovyx-sistem/

9. Задачи Data Mining. Информация и знания [Електронний ресурс] – Режим доступу до ресурсу: https://www.intuit.ru/studies/courses/6/6/lecture/164

10. Национальная библиотека им. Н. Э. Баумана Bauman National Library “Распознавание речи” [Електронний ресурс] – Режим доступу до ресурсу: https://ru.bmstu.wiki/%D0%A0%D0%B0%D1%81%D0%BF%D0%BE%D0%B7%D0%BD%D0%B0%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D1%80%D0%B5%D1%87%D0%B8

11. NLP. Основы. Техники. Саморазвитие. Часть 2: NER [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/company/abbyy/blog/449514/

12. IlovePDF [Електронний ресурс] – Режим доступу до ресурсу: https://www.ilovepdf.com/uk

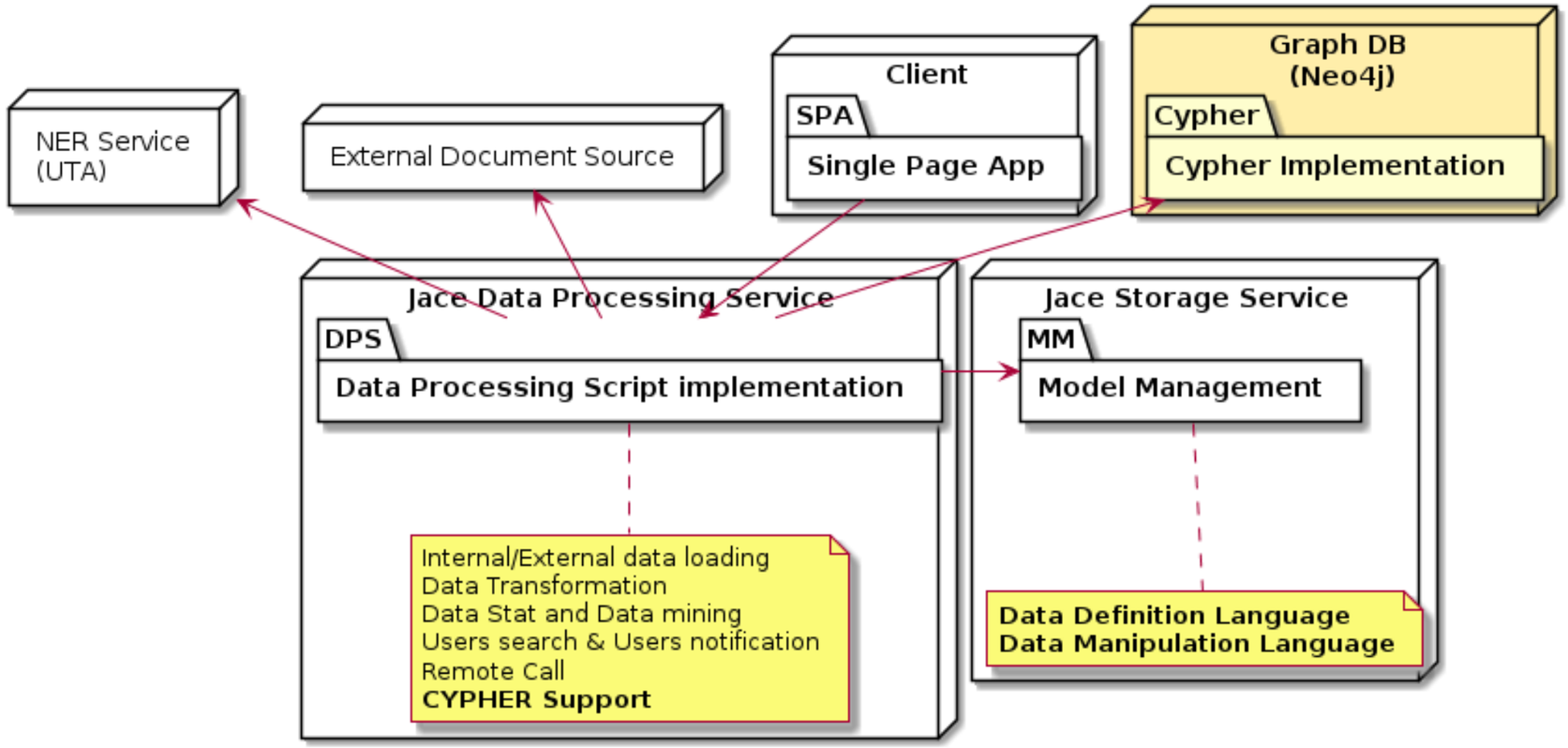
13. Семантическая информация [Електронний ресурс] – Режим доступу до ресурсу: https://www.politerm.com/zuludoc/concept_semantic.html

14. Кушнірецька О. І., Кушнірецька І. І., Берко А. Ю., 2015 “Семантичний пошук і зберігання даних науково-технічної інформаційної системи” Режим доступу до ресурсу: http://ena.lp.edu.ua:8080/bitstream/ntb/29786/1/30_310-318.pdf

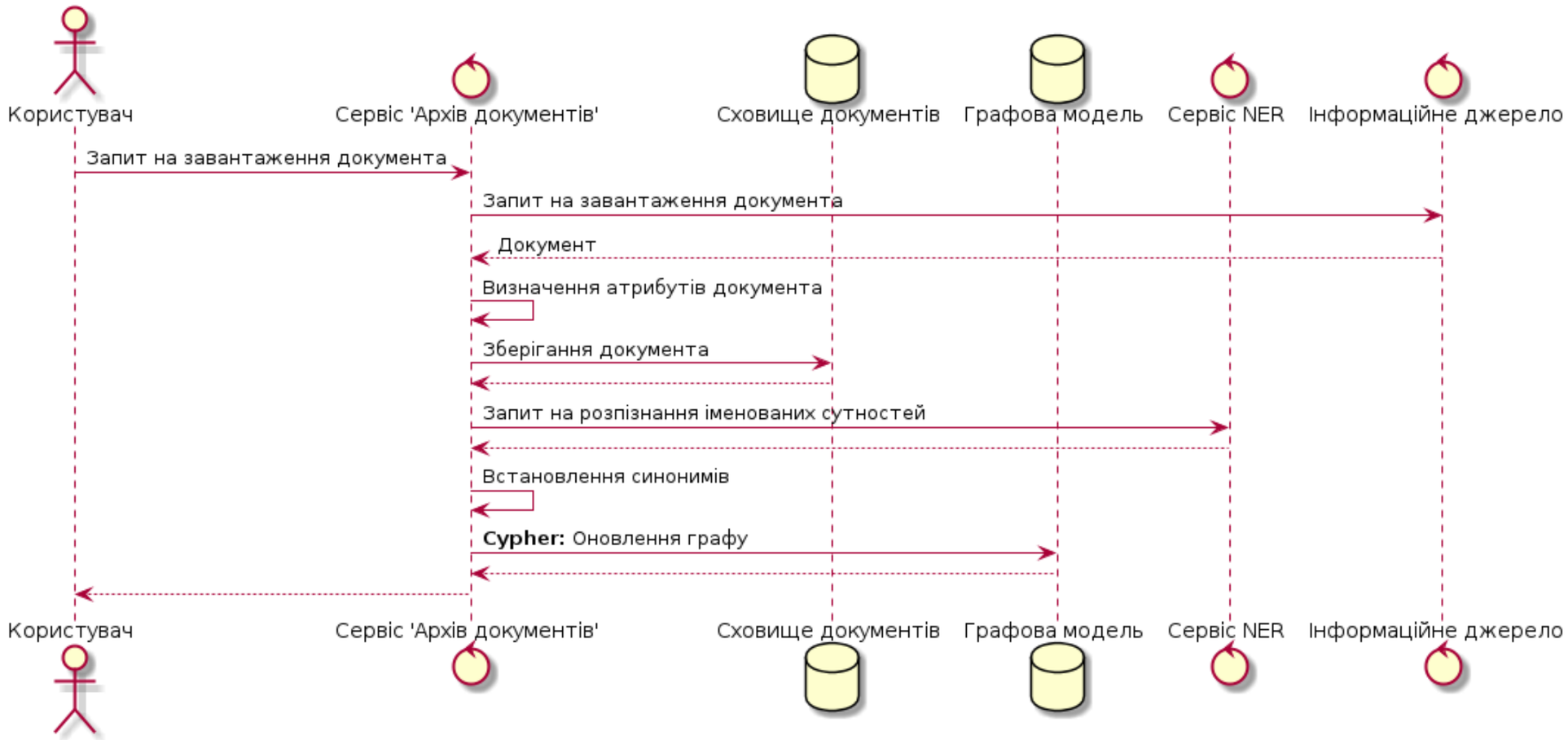
15. The Internet-Scale Graph Platform [Електронний ресурс] – Режим доступу до ресурсу: https://neo4j.com/product/?ref=home

16. JACE [Електронний ресурс] – Режим доступу до ресурсу: https://jace-dev.herokuapp.com/design/Cypher#/

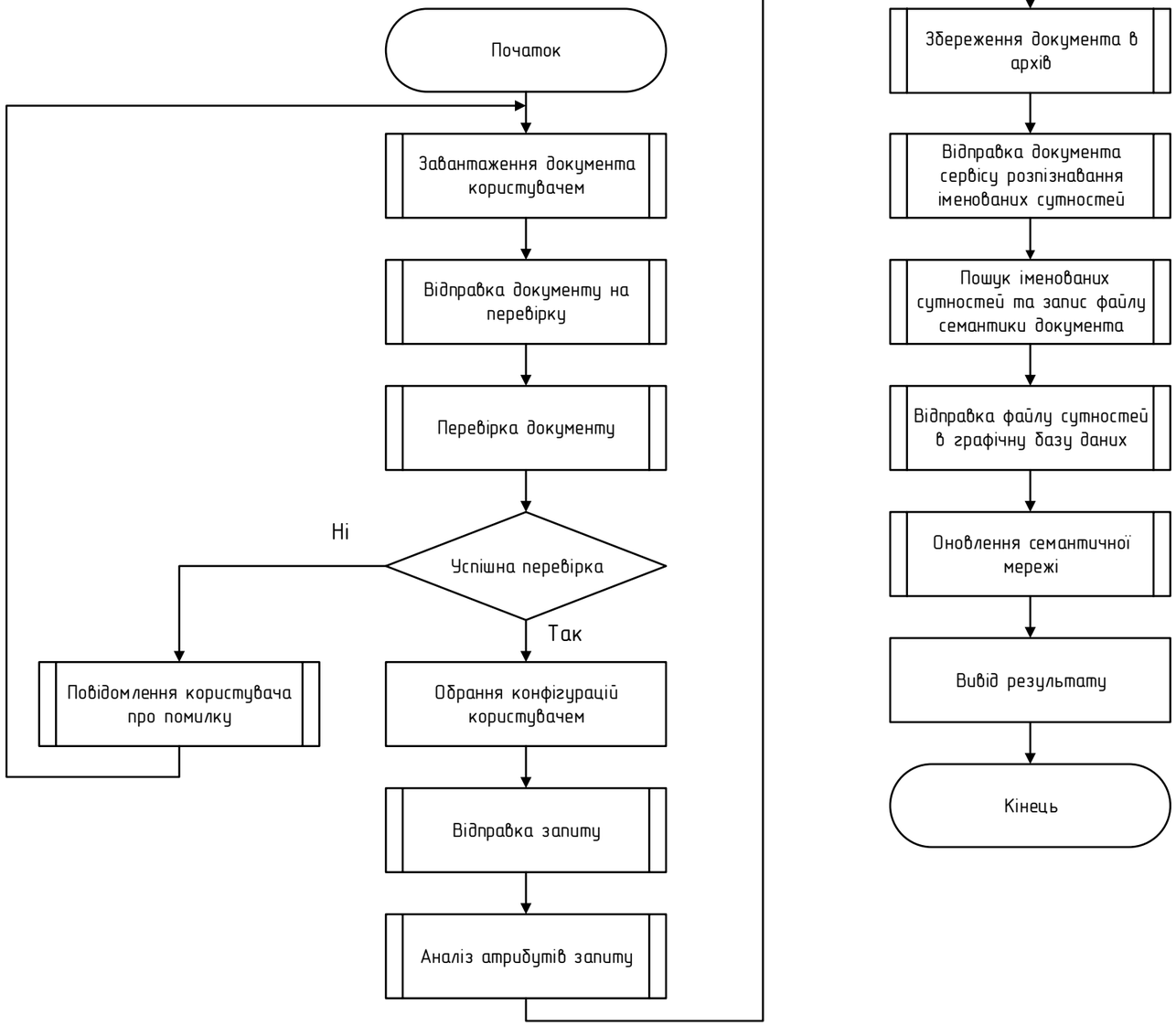
ДОДАТОК А



					ДП 6115.03.000 Д1	
					Структура модулів сервісу. Схема структурна	Літера / Масштаб
Зм.	Арк.	№ докум.	Підпис	Дата		
Розроб.		Левківський В.В.				Аркуш 1 / Аркушів 1
Перевір.		Болдак А.О.				
Т. контр.						НТУУ "КПІ" ФІОТ Група ІО-61
Н. контр.		Сімоменко В.П.				
Затв.						



					ДП 6115.04.000 Д2	
					Послідовність передачі повідомлень для завантаження документу. Схема функціональна.	Літера: — Масштаб: — Аркуш 1 Аркушів 1
Зм.	Арк.	№ докум.	Підпис	Дата		
Розроб.		Левківський В.В.				НТУУ "КПІ" ФІОТ Група ІО-61
Перевір.		Болдак А.О.				
Т. контр.						
Н. контр.		Сімоменко В.П.				
Затв.						



					ДП 6115.05.000 ДЗ	
Зм.	Арк.	№ докум.	Підпис	Дата	Опрацювання запиту обробки документа. Схема принципова	Літера / Масштаб
Розроб.		Левківський В.В.				
Перевір.		Болдак А.О.				Аркуш 1 / Аркушів 1
Т. контр.						
Н. контр.		Сімоменко В.І.				НТУУ "КПІ" ФІОТ Група ІО-61
Затв.						